Elette Boyle
Mohammad Mahmoody (Eds.)

LNCS 15365

Theory of Cryptography

22nd International Conference, TCC 2024
Milan, Italy, December 2–6, 2024
Proceedings, Part II

2 Part II

Springer

Lecture Notes in Computer Science **15365**

The series Lecture Notes in Computer Science (LNCS), including its subseries Lecture Notes in Artificial Intelligence (LNAI) and Lecture Notes in Bioinformatics (LNBI), has established itself as a medium for the publication of new developments in computer science and information technology research, teaching, and education.

LNCS enjoys close cooperation with the computer science R & D community, the series counts many renowned academics among its volume editors and paper authors, and collaborates with prestigious societies. Its mission is to serve this international community by providing an invaluable service, mainly focused on the publication of conference and workshop proceedings and postproceedings. LNCS commenced publication in 1973.

Elette Boyle · Mohammad Mahmoody
Editors

Theory of Cryptography

22nd International Conference, TCC 2024
Milan, Italy, December 2–6, 2024
Proceedings, Part II

Springer

Editors
Elette Boyle
NTT Research
Sunnyvale, CA, USA

Reichman University
Herzliya, Israel

Mohammad Mahmoody
University of Virginia
Virginia, VA, USA

ISSN 0302-9743 ISSN 1611-3349 (electronic)
Lecture Notes in Computer Science
ISBN 978-3-031-78016-5 ISBN 978-3-031-78017-2 (eBook)
https://doi.org/10.1007/978-3-031-78017-2

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

The 22nd Theory of Cryptography Conference (TCC 2024) was held during December 2–6, 2024, at Bocconi University in Milano, Italy. It was sponsored by the International Association for Cryptologic Research (IACR). The general chair of the conference was Emmanuela Orsini.

The conference received 172 submissions, of which the Program Committee (PC) selected 68 for presentation, giving an acceptance rate of 39.5%. Each submission was reviewed by at least three PC members in a single-blind process. The 50 PC members (including PC chairs), all top researchers in our field, were helped by 185 external reviewers, who were consulted when appropriate. These proceedings consist of the revised versions of the 68 accepted papers. The revisions were not reviewed, and the authors bear full responsibility for the content of their papers.

We are extremely grateful to Kevin McCurley for providing fast and reliable technical support for the HotCRP review software. We also thank Kay McKelly for her help with the conference website.

This was the tenth year that TCC presented the Test of Time Award to an outstanding paper that was published at TCC at least eight years ago, making a significant contribution to the theory of cryptography, preferably with influence also in other areas of cryptography, theory, and beyond. This year, the Test of Time Award Committee selected the following paper, published at TCC 2004: "Notions of Reducibility between Cryptographic Primitives," by Omer Reingold, Luca Trevisan, and Salil P. Vadhan. The award committee recognized this paper "for providing a rigorous and systematic taxonomy of reductions in cryptography, and in particular coining fully black-box reductions and motivating their use in barrier results."

We are greatly indebted to the many people who were involved in making TCC 2024 a success. Thank you to all the authors who submitted papers to the conference and to the PC members for their hard work, dedication, and diligence in reviewing and selecting the papers. We are also thankful to the external reviewers for their volunteered hard work and investment in reviewing papers and answering questions. Finally, thank you to the general chair Emmanuela Orsini and her team at Bocconi University, as well as to the TCC Steering Committee.

October 2024

Elette Boyle
Mohammad Mahmoody

Organization

General Chair

Emmanuela Orsini	Bocconi University, Italy

Program Committee Chairs

Elette Boyle	Reichman University, Israel & NTT Research, USA
Mohammad Mahmoody	University of Virginia, USA

Steering Committee

Yuval Ishai	Technion, Israel
Huijia (Rachel) Lin	University of Washington, USA
Tal Malkin	Columbia University, USA
Jesper Buus Nielsen	Aarhus University, Denmark
Krzysztof Pietrzak	Institute of Science and Technology Austria, Austria
Manoj M. Prabhakaran	IIT Bombay, India
Salil Vadhan	Harvard University, USA

Program Committee

Prabhanjan Ananth	UC Santa Barbara, USA
Benny Applebaum	Tel Aviv University, Israel
Amos Beimel	Ben-Gurion University of the Negev, Israel
Chris Brzuska	Aalto University, Finland
Yilei Chen	Tsinghua University, China
Ran Cohen	Reichman University, Israel
Geoffroy Couteau	CNRS, IRIF, Université Paris Cité, France
Itai Dinur	Ben-Gurion University of the Negev, Israel
Yevgeniy Dodis	New York University, USA
Stefan Dziembowski	University of Warsaw & IDEAS NCBR, Poland
Nils Fleischhacker	Ruhr University Bochum, Germany

Chaya Ganesh	Indian Institute of Science, Bangalore, India
Aarushi Goel	NTT Research, USA
Siyao Guo	NYU Shanghai, China
Mohammad Hajiabadi	University of Waterloo, Canada
Carmit Hazay	Bar-Ilan University, Israel
Justin Holmgren	NTT Research, USA
Aayush Jain	Carnegie Mellon University, USA
Zhengzhong Jin	Northeastern University, USA
Dakshita Khurana	University of Illinois at Urbana-Champaign, USA
Susumu Kiyoshima	NTT Social Informatics Laboratories, Japan
Lisa Kohl	CWI Amsterdam, Netherlands
Ilan Komargodski	Hebrew University of Jerusalem, Israel & NTT Research, USA
Eyal Kushilevitz	Technion, Israel
Huijia (Rachel) Lin	University of Washington, USA
Alex Lombardi	Princeton University, USA
Fermi Ma	Simons Institute and UC Berkeley, USA
Hemanta K. Maji	Purdue University, USA
Giulio Malavolta	Bocconi University, Italy & Max Planck Institute, Germany
Noam Mazor	Tel Aviv University, Israel
Pierre Meyer	Aarhus University, Denmark
Ryo Nishimaki	NTT Social Informatics Laboratories, Japan
Omer Paneth	Tel Aviv University, Israel
Krzysztof Pietrzak	Institute of Science and Technology Austria, Austria
Manoj Prabhakaran	IIT Bombay, India
Willy Quach	Weizmann Institute of Science, Israel
Divya Ravi	University of Amsterdam, Netherlands
Alon Rosen	Bocconi University, Italy and Reichman University, Israel
Lior Rotem	Stanford University, USA
Peter Scholl	Aarhus University, Denmark
Sruthi Sekar	IIT Bombay, India
Luisa Siniscalchi	Technical University of Denmark, Denmark
Eliad Tsfadia	Georgetown University, USA
Prashant Nalini Vasudevan	National University of Singapore, Singapore
Muthu Venkitasubramaniam	Georgetown University, USA
Mingyuan Wang	UC Berkeley, USA
Daniel Wichs	Northeastern University & NTT Research, USA
Takashi Yamakawa	NTT Social Informatics Laboratories, Japan

Additional Reviewers

Behzad Abdolmaleki
Anasuya Acharya
Amit Agarwal
Divesh Aggarwal
Andris Ambainis
Gilad Asharov
Thomas Attema
David Balbás
Laasya Bangalore
James Bartusek
Tyler Besselman
Rishabh Bhadauria
Kaartik Bhushan
Alexander Bienstock
Aniruddha Biswas
Alexander Block
Jeremiah Blocki
Katharina Boudgoust
Nicholas Brandt
Rares Buhai
Alper Cakan
Matteo Campanelli
Ran Canetti
Rutchathon Chairattana-Apirom
Benjamin Chan
Anirudh Chandramouli
Rohit Chatterjee
Megan Chen
Jessica Chen
Binyi Chen
Arka Rai Choudhuri
Sandro Coretti-Drayton
Quand Dao
Pratish Datta
Giovanni Deligios
Marian Dietz
Fangqi Dong
Nico Döttling
Ehsan Ebrahimi
Christoph Egger
Saroja Erabelli
Grzegorz Fabiański
Pooya Farshim
Giacomo Fenzi
Ben Fisch
Pouyan Forghani
Cody Freitag
Phillip Gajland
Karthik Gajulapalli
Rachit Garg
Sanjam Garg
Riddhi Ghosal
Satrajit Ghosh
Suparno Ghoshal
Niv Gilboa
Eli Goldin
Tian Gong
Junqing Gong
Jiaxin Guan
Aditya Gulati
Taiga Hiroka
Iftach Haitner
David Heath
Aditya Hegde
Hans Heum
Minki Hhan
Yao-ching Hsieh
Zihan Hu
Jihun Hwang
Yuval Ishai
Abhishek Jain
Daniel Jost
Eliran Kachlon
Fatih Kaleoglu
Chethan Kamath
Simon Kamp
Julia Kastner
Shuichi Katsumata
Hannah Keller
Hamidreza Amini Khorasgani
Taechan Kim
Elena Kirshanova
Ohad Klein
Karen Klein
Dimitris Kolonelos
Chelsea Komlo

Manu Kondapaneni
Venkata Koppula
Alexis Korb
Nishat Koti
Roman Langrehr
Seunghoon Lee
Keewoo Lee
Zeyong Li
Yunqi Li
Hanjun Li
Xiao Liang
Fuchun Lin
Chuanwei Lin
Haoxing Lin
Yao-Ting Lin
Tianren Liu
Jiahui Liu
Chen-Da Liu-Zhang
Zhenjian Lu
Donghang Lu
Vadim Lyubashevsky
Ulysse Léchine
Nir Magrafta
Bernardo Magri
Nathan Manohar
Xinyu Mao
Marcin Mielniczuk
Ethan Mook
Tomoyuki Morimae
Changrui Mu
Saachi Mutreja
Anne Müller
Varun Narayanan
Barak Nehoran
Ky Nguyen
Hai Hoang Nguyen
Guilhem Niot
Oded Nir
Aysan Nishaburi
Mahak Pancholi
Aditi Partap
Anat Paskin-Cherniavsky
Rutvik Patel
Shravani Patil
Sikhar Patranabis
Alice Pellet-Mary
Paola de Perthuis
Naty Peter
Spencer Peters
Bertram Poettering
Guru Vamsi Policharla
Alexander Poremba
Luowen Qian
Rajeev Raghunath
Debasish Ray Chawdhuri
Hanlin Ren
Doreen Riepel
Ron D. Rothblum
Adeline Roux-Langlois
Lawrence Roy
Elahe Sadeghi
Pratik Sarkar
Rahul Satish
Benjamin Schlosser
Akash Shah
Jad Silbak
Mark Simkin
Fabrizio Sisinni
Tomer Solomon
Fang Song
Katerina Sotiraki
Noah Stephens-Davidowitz
Gilad Stern
Björn Tackmann
Kel Zin Tan
Er-cheng Tang
Athina Terzoglou
Jean-Pierre Tillich
Pratyush Ranjan Tiwari
Daniel Tschudi
Prashant Vasudevan
Ivan Visconti
Benedikt Wagner
William Wang
Benjamin Wesolowski
Jiawei Wu
David Wu
Yu Xia
Zhiye Xie
Jeff Xu

Anshu Yadav
Sophia Yakoubov
Chao Yan
Yibin Yang
Xiuyu Ye
Eylon Yogev
Albert Yu
Ilias Zadik
Runzhi Zeng

Contents – Part II

Quantum I

Math & Foundations

Proofs II

Quantum I

Quantum Pseudorandom Scramblers

Chuhan Lu[1(✉)], Minglong Qin[2], Fang Song[1], Penghui Yao[2,3], and Mingnan Zhao[2]

[1] Computer Science Department, Portland State University, Portland, USA
{chuhan,fang.song}@pdx.edu
[2] State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China
mlqin,zhao@smail.nju.edu.cn
[3] Hefei National Laboratory, Hefei 230088, China

Abstract. Quantum pseudorandom state generators (PRSGs) have stimulated exciting developments in recent years. A PRSG, on a fixed initial (e.g., all-zero) state, produces an output state that is computationally indistinguishable from a Haar random state. However, pseudorandomness of the output state is not guaranteed on other initial states. In fact, known PRSG constructions *provably* fail on some initial states.

In this work, we propose and construct quantum Pseudorandom State Scramblers (PRSSs), which can produce a pseudorandom state on an *arbitrary* initial state. In the information-theoretical setting, we obtain a scrambler which maps an arbitrary initial state to a distribution of quantum states that is close to Haar random in *total variation distance*. As a result, our scrambler exhibits a *dispersing* property. Loosely, it can span an ϵ-net of the state space. This significantly strengthens what standard PRSGs can induce, as they may only concentrate on a small region of the state space provided that the average output state approximates a Haar random state.

Our PRSS construction develops a *parallel* extension of the famous Kac's walk, and we show that it mixes *exponentially* faster than the standard Kac's walk. This constitutes the core of our proof. We also describe a few applications of PRSSs. While our PRSS construction assumes a post-quantum one-way function, PRSSs are potentially a weaker primitive and can be separated from one-way functions in a relativized world similar to standard PRSGs.

Keywords: Quantum pseudorandom states · Kac's walk · Pseudorandom unitary operators

1 Introduction

Pseudorandomness is a fundamental concept in complexity theory and cryptography, offering efficient approximation to true randomness against computationally bounded adversaries. Recently, Ji, Liu and Song [30] introduced quantum pseudorandom state generators (PRSGs) as a family of quantum states

E. Boyle and M. Mahmoody (Eds.): TCC 2024, LNCS 15365, pp. 3–35, 2025.
https://doi.org/10.1007/978-3-031-78017-2_1

$\{|\phi_k\rangle\}_{k\in\mathcal{K}}$, which can be generated in polynomial time, and no computationally-bounded quantum adversary can distinguish polynomially many copies of $|\phi_k\rangle$ from polynomially many copies of a Haar random state. PRSGs can be considered as a quantum counterpart to classical pseudorandom generators, and can be constructed assuming the existence of one-way functions that are hard for efficient quantum adversaries [1,4,9,10,30]. What is surprising, PRSGs are proven weaker than one-way functions in a relativized world [34,35]. Since one-way functions are considered the *minimal* assumption in classical cryptography, this opens up the possibility of basing quantum cryptography on *weaker* assumptions. There have been exciting advances in recent years, realizing a host of cryptographic tasks based on PRSGs [4–6,18,38]. In addition to cryptographic interest, pseudorandom states have also inspired new developments for quantum gravity theory and string theory [1,7,11,33,45].

Another fundamental quantum pseudorandom primitive, *pseudorandom unitary operators* (PRUs), was also introduced in [30] as a quantum analogue of pseudorandom functions. A PRU is a set of polynomially-time unitary operators that are computationally indistinguishable from Haar random unitaries. PRUs clearly imply PRSGs and could further enrich the toolkit in cryptography and physics [7,11,21,33,45]. Nonetheless, constructing a provably-secure PRU remains an open problem, and progress has been slow (e.g., conjectured constructions in [30], a stateful simulation in [2], and on the negative side some barriers such as impossibility of PRUs that are sparse or of real entries [26]). In fact, even basic properties that are *necessary* for PRUs have not been achieved. It is easy to see that a PRU gives a family of polynomial-sized quantum circuits which can map an *arbitrary* pure state to a family of pseudorandom states. However, a PRSG can be viewed as a family of polynomial-sized quantum circuits which map a specific initial state, typically $|0^n\rangle$, to a family of pseudorandom states. Indeed, all existing constructions of PRSGs necessitate a specific initial state, and it can be shown that they *fail* to produce pseudorandom states for certain initial states. This limitation has indeed caused a variety of technical challenges in the cryptographic applications mentioned before that need to be addressed in ad hoc ways. It hence becomes imperative to understand the following question and its consequences.

Can we construct a family of polynomial-sized quantum circuits which map an arbitrary input (pure) state to pseudorandom states?

1.1 Our Contributions

In this work, we answer the question affirmatively as a steady step towards bridging the gap between PRSGs and PRUs. We formally encapsulate the property of "scrambling" an arbitrary input state in a novel quantum pseudorandom primitive, termed a *quantum pseudorandom state scrambler* (PRSS), which isometrically maps an *arbitrary* pure state to a pseudorandom state. We then construct a PRSS based on any quantum-secure PRF. A central technical novelty is to design a *parallel* version of Kac's walk, which is a random walk on a unit

sphere, and prove a mixing time *exponentially* faster than the standard Kac's walk [42]. Although Kac's walk was introduced by Kac in [32] more than half a century ago and has been studied by a large body of works since then, this work, to our knowledge, is the first time to employ Kac's walk to design quantum pseudorandom objects.

Our construction also exhibits a notable *dispersing* property. Loosely speaking, the output states of our scrambler constitute an ϵ-net on the sphere, and the distribution closely approximates the Haar random distribution under the strong Wasserstein distance, when sufficient randomness is supplied. Such a powerful "randomizing" capability needs not be present even in PRUs.

Overview on the Construction and Analysis. Our construction is inspired by Kac's walk, originally a model for a Boltzmann gas [32]. This approach differs from previous constructions for PRSGs. Let us consider an arbitrary unit-vector $v \in \mathbb{R}^N$. In one step of Kac's walk, two distinct coordinates (i, j) and an angle $\theta \in [0, 2\pi)$ are chosen uniformly at random. Then $R_\theta := \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$ is applied to rotate the two-dimensional subvector $(v_i, v_j)^T$. It is proven that it converges to the Haar measure on the unit sphere of $\mathbb{R}^N$ in $O(N \log N)$ steps [42]. However, if we view the input vector as an $n = \log N$-qubit state, then the factor N in the mixing time is prohibitive for the purpose of an efficient (polynomial in n) scrambler.

Can we *parallelize* Kac's walk in hope of shaking off a factor of N? Notice that in Kac's walk, if any two consecutive steps overlap on the random choices of coordinates, then they need to be executed in *sequence*. One might consider conditioning on the event of "collision-free" in the coordinate choices, but this occurs with negligibly small probability since we intend to compress $\Omega(N)$ steps into one.

We design a *parallel* Kac's walk that rapidly mixes in $O(\log N)$ time, an *exponential* improvement over the original walk. In each step, instead of working with an individual pair of coordinates, we randomly partition the N coordinates into $N/2$ pairs, and then each pair is rotated by a random angle chosen *independently*. Although the mixing time of Kac's walk is not directly applicable, we show that the specific path-coupling proof strategy of [42] can be extended here.

We then construct a quantum circuit to implement our parallel Kac's walk. In each step, we use a random permutation to realize the coordinate partition, and employ a random function to compute a random rotation angle, under a careful discretization, for each pair of coordinates. Finally, we obtain our pseudorandom state scramblers by replacing the random permutations and functions with quantum-secure pseudorandom permutations and functions, which exist based on post-quantum one-way functions [47].

The discussion so far works with real Hilbert spaces. To construct a PRSS in a complex Hilbert space, we further develop a parallel Kac's walk on complex Hilbert spaces. The construction starts likewise by randomly partitioning N coordinates to $N/2$ pairs, and then applying random 2×2 *unitary* matrices

independently to each pair. As unitary matrices have more degrees of freedom than real orthogonal rotation matricies, the analysis of the mixing time is more involved. The extension of Kac's walk to a complex Hilbert space, as well as the parallelization, has not been studied previously as far as we are aware. This may be of independent interest. Due to space constraints, this paper focuses on the real case. A comprehensive treatment of the complex case is provided in the full version [36].

Applications. It is easy to see that PRSSs subsume standard PRSGs as well as scalable PRSGs. We also demonstrate that PRSSs can be used to achieve a black-box realization of a variant of PRSGs known as pseudorandom function-like state generators (PRFSGs), which in turn enable a host of cryptographic primitives such as IND-CPA SKE and EUF-CMA MAC [4,5]. A PRFSG takes an additional classical input x (from a poly-size domain) and produces a pseudorandom state. In the literature, a PRFSG (with logarithmic input length) can be constructed from PRSGs by measuring a part of a pseudorandom state and then post-selecting on x. This inevitably is error-prone and consumes multiple copies, i.e., multiple invocations of a PRSG, to evaluate on a single x. Given our PRSS (with a sufficiently long key), we can simply feed $|x\rangle$ as the initial state to the PRSS, and hence only *one*, rather than polynomially-many, run of PRSS suffices.

We observe that the argument by Kretschmer [34] also implies that PRSS is *strictly* weaker than one-way functions relative to an oracle. Thus PRSSs may further enhance the new cryptographic landscape without assuming one-way functions. We demonstrate some use cases of PRSSs beyond what are already possible from PRSGs. For starters, a PRSS enables efficient encryption of quantum messages by effectively "scrambling" any initial state, and allowing *multiple copies* of the same state to be encrypted under the same key. The fact that PRSS provides a secure encryption also enables committing quantum states, thanks to a new characterization of [21]. The commitment scheme can be further made *succinct*, where the commitment message has smaller size than the size of the message to be committed. Existing constructions rely on potentially stronger assumptions than PRSSs.

Subsequent Work

A follow-up work [3] gave a construction that is indistinguishable from applying the tensor product of a Haar random isometry when the input state is restricted to one of three special families: (1) $|\psi\rangle^{\otimes q}$ for a pure state $|\psi\rangle$ and polynomially-bounded q; (2) $\bigotimes_{i=1}^{q} |x_i\rangle$; and (3) $\otimes_{i=1}^{q} |\phi_i\rangle$, where every ϕ_i is Haar random. Their construction requires adding an ancilla system $|0^m\rangle$, and the security loss scales with $1/2^m$. As a result, it *necessarily* cannot preserve the input dimension and m is chosen to be a polynomial to obtain negligible security loss. This also incurs poly overheads in the applications such as quantum encryption. In other words, it only (unitarily) scrambles states $|\psi\rangle\,|0^m\rangle$ for a $|\psi\rangle$ chosen from

one of the three families above and a polynomial m. More recently, several independent works on constructing pseudorandom unitaries which are secure against non-adaptive queries have been presented. Metger, Poremba, Sinha and Yuen [37] proposed a construction using a composition of Clifford gates, pseudorandom functions and pseudorandom permutations. Brakerski and Magrafta [8] presented a construction for real-valued unitaries that look like Haar random on any polynomial-sized set of orthogonal input states. Chen, Bouland, Brandao, Docter, Hayden, and Xu [17] achieved similar results via products of exponentiated sums of random permutations with random phases. It is not clear whether these constructions are able to generate an ϵ-net and realize the *dispersing* property (details in Sect. 5.4), a strong randomizing property achieved by our construction.

1.2 Discussions and Open Questions

There is a rich history of studying Kac's walk in probability and mathematical physics [20,25,27,29,31,41]. Determining the total variation mixing time of Kac's walk is particularly challenging, and it is currently only known to be between the order $O(n^4 \log n)$ and $O(n^2)$ [43].

There has also been extensive efforts on approximations to Haar measures in a *statistical* setting, known as state and unitary t-designs [19,44]. For instance, a unitary t-design mimics a Haar random unitary up to the t-th moment. It is known that a unitary t-design can be constructed by a quantum circuit of size polynomially in t, composed of Haar random single or two-qubit gates [12, 22–24,40]. It is interesting to note that a path-coupling technique in [41] for analyzing Kac's walk also plays an essential role in the proofs of these unitary design results. It is reasonable to anticipate improvements on the efficiency of the unitary designs with new advances on Kac's walk. However, it is worth stressing that another critical component in their proofs involving spectral gaps appears to inevitably incur a dependency on t, which is a serious limitation. For instance, in order for the output state to approximate a Haar random state when the number of copies can be an arbitrary polynomial, we would need to pick a superpolynomial t in the unitary design. As far as we know, our PRSS is the first to employ Kac's walk directly in the construction of a quantum pseudorandom object, and the exponential improvement on the mixing time of our parallel walk enables flipping the quantifiers, i.e., a fixed poly-size construction that is nonetheless pseudorandom against any polynomial-time distinguisher, a desired feature towards PRUs.

Kac's walk has also found applications in algorithm design. Recently, a fast and memory-optimal dimension-reduction algorithm is proposed based on Kac's walk and its discrete variants [28]. We would like to invite more exploration of Kac's walk in theoretical computer science broadly.

We describe several interesting open problems emerged from our work.

1. Is it possible to simplify the quantum circuits for these primitives? Can we replace random permutations by a sequence of parallel (pseudo) random local

permutations? Can we use the same random rotation or even a fixed one (e.g., Hadamard transform) in a single iteration? Recent advances on repeated averages on graphs [39] and orthogonal repeated averaging [16,28] allude to an affirmative answer.
2. We believe that PRSSs, potentially weaker than PRUs, are an important primitive in its own right. Can we discover more applications of PRSSs and the dispersing property, especially in cryptography as well as in quantum gravity theory? For example, we envision a form of *uncloneable knowledge tokens* from a PRSS that may enable novel quantum proof systems and *delegated* computation.
3. Is our construction of PRSS capable of scrambling polynomial quantum states? This appears to require strengthening the coupling technique in our current analysis, and it might be useful to analyze other variants of Kac's walk.
4. How far are we from a PRU? Can we get it by strengthening our parallel Kac's walk approach or can we show that our construction is already a PRU? By a simple hybrid argument, it suffices to prove that our parallel Kac's walk on $\mathrm{SO}(N)$ converges within $\mathrm{polylog}(N)$ time in terms of the L^∞ Wasserstein distance. Indeed, there has been a large body of work devoted to studying the speed of the convergence with respect to different metrics [20,31,32,43]. One of the most relevant works is Oliveira's result [41] showing a tight convergence time of order $O(N^2 \log N)$ with respect to the stronger L^2 Wasserstein distance. Our parallelization achieves a quadratic speedup, which leads to an $\tilde{O}(2^n)$-time construction of PRU. Since the L^∞ Wasserstein distance is a less stringent metric than the L^2 Wasserstein distance, there is hope of obtaining an improved convergence rate. To our knowledge, the speed of convergence of Kac's walk with respect to L^∞ Wasserstein distance has not been studied, and hence developing new techniques to overcome the tightness of Oliveira's L^2 result would be an exciting research direction.

Organization. Section 2 contains preliminary materials on basic notations and cryptographic primitives. Section 3 describes definitions and properties of our new primitives. Section 4 introduces the parallel Kac's walk. Then Sect. 5 constructs PRSSs via implementing the parallel Kac's walk and introduces the dispersing RSS. Section 6 describes applications of PRSSs. Some proofs are deferred to the full version of this paper [36].

2 Preliminary

2.1 Basic Notation

For $n \in \mathbb{N}$, $[n]$ denotes $\{1, \ldots, n\}$. For $x \in \{0,1\}^n$, we use x_i to denote the i-th bit of x and define $\mathrm{val}(x) = \sum_{i=1}^{n} 2^{-i} x_i$. Suppose that x and y are bit strings of finite length, we denote xy to be the concatenation of x and y. For finite sets $\mathcal{X}$ and $\mathcal{Y}$, we use $\mathcal{X}^{\mathcal{Y}}$ to denote the set of all functions $\{f : \mathcal{X} \to \mathcal{Y}\}$. We use $S_{\mathcal{X}}$

to denote the *permutation group* over elements in a finite set $\mathcal{X}$. We often write S_{2^n} instead of $S_{\{0,1\}^n}$ to denote the permutation group over elements in $\{0,1\}^n$.

For any symbol x and $n \in \mathbb{N}$, $(x_i)_{i=1}^n$ represents $(x_1, \ldots, x_n)$. With a slight abuse of notation, we let $(x_i)_{i=1}^n \subseteq S$ represent $x_i \in S$ for all $i \in [n]$. For $n \in \mathbb{N}$, $\mathcal{S}_{\mathbb{R}}^n$ denotes the set of all unit vectors in $\mathbb{R}^n$, $\mathcal{S}_{\mathbb{C}}^n$ denotes the set of all unit vectors in $\mathbb{C}^n$, $\mathrm{SO}(n)$ denotes the special orthogonal group of $n \times n$ real matrices, $\mathrm{SU}(n)$ denotes the special unitary group of $n \times n$ complex matrices, $\mathrm{O}(n)$ denotes the $n \times n$ orthogonal group and $\mathrm{U}(n)$ denotes the $n \times n$ unitary group. For a Hilbert space $\mathcal{H}$, we use $\mathcal{S}(\mathcal{H})$ to denote the set of pure quantum states in $\mathcal{H}$ and $\mathcal{D}(\mathcal{H})$ to denote the set of density operators on $\mathcal{H}$.

For an n-dimensional vector v and $i \in [n]$, we use $v[i]$ to denote the i-th coordinate of v. For $S \subseteq [n]$ and $v \in \mathbb{C}^n$, define

$$\|v\|_1 = \sum_{i\in[n]} |v[i]|, \|v\|_{1,S} = \sum_{i\in S} |v[i]|, \|v\|_2 = \sqrt{\sum_{i\in[n]} |v[i]|^2}.$$

For an $n \times n$ matrix M and $p \in \mathbb{N}$, the p-norm of M is defined to be $\|M\|_p = \left(\mathrm{Tr}\left[\left(M^\dagger M\right)^{p/2}\right]\right)^{1/p}$, and $\|M\|_\infty$ is defined to be the largest singular value of M. The following fact will be used in our paper and is easy to prove by the triangle inequality.

Fact 1. *Given $m, n \in \mathbb{N}$, $U_1, \ldots, U_m, V_1, \ldots, V_m \in \mathrm{O}(n)$ (or $\mathrm{U}(n)$), then*

$$\|U_1 \ldots U_m - V_1 \ldots V_m\|_\infty \leq \sum_{i=1}^m \|U_i - V_i\|_\infty .$$

Given two density operators $\rho, \sigma \in \mathcal{D}(\mathcal{H})$, the trace distance between ρ and σ is $\mathrm{TD}(\rho, \sigma) = \|\rho - \sigma\|_1$.

Let $\mathcal{V}$ be a real or complex vector space, and $\epsilon > 0$ be a positive real number. For any $\mathcal{S} \subseteq V$, a set of vectors $\mathcal{N} \subseteq \mathcal{S}$ is said to be an *ϵ-net* of $\mathcal{S}$ if, for every vector $u \in \mathcal{S}$, there exists a vector $v \in \mathcal{N}$ such that $\|u - v\|_2 \leq \epsilon$.

We adopt the standard quantum circuit model. A quantum circuit with gates drawn from a finite gate set can be encoded as a binary string. $\{Q_\lambda : \lambda \in \mathbb{N}\}$ is said to be a *polynomial-time* generated family[1] if there exists a deterministic Turing machine that, on any input $\lambda \in \mathbb{N}$, outputs an encoding of Q_λ in polynomial-time in λ. A quantum polynomial-time algorithm is identified with a polynomial-time generated circuit family. In cryptography it is conventionally to model adversaries as *non-uniform* algorithms. We model a non-uniform quantum polynomial-time algorithm as a family $\{Q_\lambda, \rho_\lambda\}_\lambda$, where $\{Q_\lambda\}$ is a polynomial-time generated circuit family, and $\{\rho_\lambda\}$ is a collection of *advice states*. Q_λ acts on ρ_λ besides the actual input state.

[1] More precisely, each circuit should be written as Q_{1^λ}. Note that in a polynomial-time generated family, then Q_λ must have size polynomial in λ.

2.2 Probability Theory

For two probability measures ν_1 and ν_2 defined on measurable space $(\Omega, \mathcal{F})$, the *total variation distance* of ν_1 and ν_2 is defined as

$$\|\nu_1 - \nu_2\|_{\mathrm{TV}} = \sup_{A \in \mathcal{F}} |\nu_1(A) - \nu_2(A)| .$$

Closeness in total variation distance is a strong promise. For example, when applied to quantum states, it implies closeness in trace distance of the average states.

Lemma 1. *Let μ and ν be two arbitrary probability measures over $\mathcal{S}_{\mathbb{R}}^{2^n}$ ($\mathcal{S}_{\mathbb{C}}^{2^n}$). Then for all $\ell \in \mathbb{N}$,*

$$\left\| \mathop{\mathbb{E}}_{|\psi\rangle \sim \mu} \left[(|\psi\rangle\langle\psi|)^{\otimes \ell} \right] - \mathop{\mathbb{E}}_{|\varphi\rangle \sim \nu} \left[(|\varphi\rangle\langle\varphi|)^{\otimes \ell} \right] \right\|_1 \leq \|\mu - \nu\|_{\mathrm{TV}} .$$

We denote the distribution of a random variable X by $\mathcal{L}(X)$. If $\mathcal{L}(X) = \nu$, we write $X \sim \nu$. A coupling of two probability measures μ and ν is a joint probability measure whose marginals are μ and ν. We use $\Gamma(\mu, \nu)$ to denote the set of all couplings of μ and ν. For $p \geq 1$ The *Wasserstein p-distance* between two probability measures μ and ν is

$$W_p(\mu, \nu) = \left(\inf_{\gamma \in \Gamma(\mu,\nu)} \mathop{\mathbb{E}}_{(x,y)\sim\gamma} [\|x - y\|_2^p] \right)^{1/p} .$$

The Wasserstein ∞-distance is $W_\infty(\mu, \nu) = \lim_{p \to \infty} W_p(\mu, \nu)$.

2.3 Cryptography

In this section, we will review various definitions and results in cryptography. Throughout this work, λ denotes a security parameter.

Pseudorandom Functions and Pseudorandom Permutations.

Definition 1 (Quantum-Secure Pseudorandom Function). *Let $\mathcal{K}, \mathcal{X}$ and $\mathcal{Y}$ be the key space, the domain and range, all implicitly depending on the security parameter λ. A keyed family of functions $\{\mathsf{PRF}_k : \mathcal{X} \to \mathcal{Y}\}_{k \in \mathcal{K}}$ is a quantum-secure pseudorandom function (QPRF) if the following two conditions hold:*

1. ***Efficient generation.*** *PRF_k is polynomial-time computable on a classical computer.*
2. ***Pseudorandomness.*** *For any polynomial-time quantum oracle algorithm $\mathcal{A}$, PRF_k with a random $k \leftarrow \mathcal{K}$ is indistinguishable from a truly random function $f \leftarrow \mathcal{Y}^{\mathcal{X}}$ in the sense that:*

$$\left| \Pr_{k \leftarrow \mathcal{K}} \left[\mathcal{A}^{\mathsf{PRF}_k}(1^\lambda) = 1 \right] - \Pr_{f \leftarrow \mathcal{Y}^{\mathcal{X}}} \left[\mathcal{A}^{f}(1^\lambda) = 1 \right] \right| = \mathrm{negl}(\lambda) .$$

Definition 2 (Quantum-Secure Pseudorandom Permutation). *Let $\mathcal{K}$ be the key space, and $\mathcal{X}$ be both the domain and range, implicitly depending on the security parameter λ. A keyed family of permutations $\{\mathsf{PRP}_k \in S_{\mathcal{X}}\}_{k\in\mathcal{K}}$ is a quantum-secure pseudorandom permutation (*QPRP*) if the following two conditions hold:*

1. *(**Efficient generation**). PRP_k and PRP_k^{-1} are polynomial-time computable on a classical computer.*
2. *(**Pseudorandomness**). For any polynomial-time quantum oracle algorithm $\mathcal{A}$, PRP_k with a random $k \leftarrow \mathcal{K}$ is indistinguishable from a truly random permutation $\sigma \leftarrow S_{\mathcal{X}}$ in the sense that:*

$$\left| \Pr_{k\leftarrow\mathcal{K}}\left[\mathcal{A}^{\mathsf{PRP}_k,\mathsf{PRP}_k^{-1}}\left(1^\lambda\right)=1\right] - \Pr_{\sigma\leftarrow S_{\mathcal{X}}}\left[\mathcal{A}^{\sigma,\sigma^{-1}}\left(1^\lambda\right)=1\right]\right| = \mathrm{negl}(\lambda)\,.$$

We adopt the definition of a strong quantum-secure PRP in this paper. And when referring to a quantum oracle algorithm having oracle access to a permutation σ, we imply that it has oracle access to both σ and its inverse σ^{-1}.

Under the assumption that post-quantum one-way functions exist, Zhandry proved the existence of QPRFs [47]. QPRPs can be constructed from QPRFs efficiently [46].

Given two QPRFs F and G, one independently samples F_{k_1} from F and G_{k_2} from G. A standard hybrid argument shows that F_{k_1}, G_{k_2} are computationally indistinguishable from two independent random functions, as stated in the following lemma. The proof, detailed in the full version of this paper, can be readily extended to the scenario when polynomially many pseudorandom primitives (or random primitives) are given.

Lemma 2. *Let keyed families of functions $F : \mathcal{K}_1\times\mathcal{X}_1 \to \mathcal{Y}_1$ and $G : \mathcal{K}_2\times\mathcal{X}_2 \to \mathcal{Y}_2$ be* QPRF*s. Then we have for any polynomial-time quantum oracle algorithm $\mathcal{A}$,*

$$\left| \Pr_{k_1\leftarrow\mathcal{K}_1,k_2\leftarrow\mathcal{K}_2}\left[\mathcal{A}^{F_{k_1},G_{k_2}}\left(1^\lambda\right)=1\right] - \Pr_{f\leftarrow\mathcal{Y}_1^{\mathcal{X}_1},g\leftarrow\mathcal{Y}_2^{\mathcal{X}_2}}\left[\mathcal{A}^{f,g}\left(1^\lambda\right)=1\right]\right| = \mathrm{negl}(\lambda)\,.$$

It also holds if $\mathcal{X}_2 = \mathcal{Y}_2$, G is a family of QPRP*s and $g \leftarrow \mathcal{Y}_2^{\mathcal{X}_2}$ is replaced by $g \leftarrow S_{\mathcal{X}_2}$.*

Quantum Pseudorandomness. The concept of quantum pseudorandom state generators was originally introduced in [30].

Definition 3 (Quantum Pseudorandom State Generator). *Let $\mathcal{K}$ be a key space and $\mathcal{H}$ be a Hilbert space. $\mathcal{K}$ and $\mathcal{H}$ depend on the security parameter λ. A pair of polynomial-time quantum algorithms (K, G) is a pseudorandom state generator (*PRSG*) if the following holds:*

- ***Key Generation.** $K(1^\lambda)$ chooses a uniform $k \in \mathcal{K}$ and outputs it as the key.*

- ***State Generation.*** *For all $k \in \mathcal{K}$, $G(1^\lambda, k)$ outputs a quantum state $|\phi_k\rangle \in \mathcal{S}(\mathcal{H})$.*
- ***Pseudorandomness.*** *Any polynomially many copies of $|\phi_k\rangle$ with the same random k is computationally indistinguishable from the same number of copies of a Haar random state. More precisely, for any $n \in \mathbb{N}$, any efficient quantum algorithm $\mathcal{A}$ and any $\ell \in \mathrm{poly}(\lambda)$,*

$$\left| \Pr_{k \leftarrow \mathcal{K}} \left[\mathcal{A}\left(|\phi_k\rangle^{\otimes \ell} \right) = 1 \right] - \Pr_{|\psi\rangle \leftarrow \mu} \left[\mathcal{A}\left(|\psi\rangle^{\otimes \ell} \right) = 1 \right] \right| = \mathrm{negl}(\lambda)\,,$$

where μ is the Haar measure on $\mathcal{S}(\mathcal{H})$.

*We call the keyed family of quantum states $\{\phi_k\}_{k\in\mathcal{K}}$ a pseudorandom quantum state (*PRS*) in $\mathcal{H}$.*

PRSGs exist assuming the existence of QPRFs. Given any QPRF $\mathsf{PRF} : \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ (where $\mathcal{K}$ and $N = 2^n$ are implicitly functions of the security parameter λ), [30] constructed a PRS $\{\phi_k\}_{k\in\mathcal{K}}$, referred to *(pseudo)random phase states*, as follows:

$$|\phi_k\rangle = \frac{1}{\sqrt{N}} \sum_{x\in\{0,1\}^n} \omega_N^{\mathsf{PRF}_k(x)} |x\rangle$$

for $k \in \mathcal{K}$ and $\omega_N = e^{\mathrm{i}\frac{2\pi}{N}}$. Additionally, they conjectured the variant with binary phase (i.e., replacing ω_N with -1) remains a PRS, and this was later confirmed in [9].

It is worth noting that both of these constructions rely on state generation algorithms that require a specific initial state, typically the all-zero state $|0\rangle^{\otimes n}$. If we were to use a different initial state, such as the equally weighted superposition state $|+\rangle^{\otimes n}$, their state generation algorithms would fail to produce a pseudorandom state. Therefore, the specific initial state is crucial for the success of these constructions.

3 Pseudorandom State Scramblers

We describe our new primitive *quantum Pseudorandom State Scramblers* (PRSS). A PRSS is capable of generating a pseudorandom state on an arbitrary initial state, addressing the limitation of acting on one specific initial state.

Definition 4 (Pseudorandom State Scrambler). *Let $\mathcal{H}_{\mathrm{in}}$ and $\mathcal{H}_{\mathrm{out}}$ be Hilbert spaces of dimensions 2^n and 2^m respectively with $n, m \in \mathbb{N}$ and $n \leq m$. Let $\mathcal{K} = \{0,1\}^\kappa$ be a key space, and λ be a security parameter. A* pseudorandom state scrambler *(*PRSS*) is an ensemble of isometric operators*

$$\mathcal{R}^{n,m} := \{\{\mathcal{R}_k^{n,m,\lambda} : \mathcal{H}_{\mathrm{in}} \to \mathcal{H}_{\mathrm{out}}\}_{k\in\mathcal{K}}\}_\lambda,$$

satisfying:

- ***Pseudorandomness***. *For any* $\ell = \mathrm{poly}(\lambda)$, *any* $|\phi\rangle \in \mathcal{S}(\mathcal{H}_{\mathrm{in}})$, *and any non-uniform poly-time quantum adversary* $\mathcal{A}$,

$$\left| \Pr_{k \leftarrow \mathcal{K}} \left[\mathcal{A}\left(|\phi_k\rangle^{\otimes \ell} \right) = 1 \right] - \Pr_{|\psi\rangle \leftarrow \mu} \left[\mathcal{A}\left(|\psi\rangle^{\otimes \ell} \right) = 1 \right] \right| = \mathrm{negl}(\lambda)\,,$$

  where $|\phi_k\rangle := \mathcal{R}_k^{n,m,\lambda} |\phi\rangle$ *and* μ *is the Haar measure on* $\mathcal{S}(\mathcal{H}_{\mathrm{out}})$.
- ***Uniformity***. $\mathcal{R}^{n,m}$ *can be uniformly computed in polynomial time. That is, there is a deterministic Turing machine that, on input* $(1^n, 1^m, 1^\lambda, 1^\kappa)$, *outputs a quantum circuit* Q *in* $\mathrm{poly}(n, m, \lambda, \kappa)$ *time such that for all* $k \in \mathcal{K}$ *and* $|\phi\rangle \in \mathcal{S}(\mathcal{H}_{\mathrm{in}})$

$$Q\,|k\rangle|\phi\rangle = |k\rangle|\phi_k\rangle\,,$$

  where $|\phi_k\rangle := \mathcal{R}_k^{n,m,\lambda}|\phi\rangle$.
- ***Polynomially-bounded key length***. $\kappa = \log|\mathcal{K}| = \mathrm{poly}(m, \lambda)$. *As a result,* $\mathcal{R}^{n,m}$ *can be computed efficiently in time* $\mathrm{poly}(n, m, \lambda)$.

By strengthening the pseudorandomness condition in PRSS, we define random state scramblers as follows. We also present a table (Table 1) that compares our new primitives with other quantum (pseudo) random objects.

Definition 5 (Random State Scrambler). *Let* $\mathcal{H}_{\mathrm{in}}$ *and* $\mathcal{H}_{\mathrm{out}}$ *be Hilbert spaces of dimensions* 2^n *and* 2^m *respectively with* $n, m \in \mathbb{N}$ *and* $n \leq m$. *Let* $\mathcal{K} = \{0,1\}^\kappa$ *be a key space, and* λ *be a security parameter. A* random state scrambler *(*RSS*) is an ensemble of isometric operators* $\mathcal{R}^{n,m} := \{\mathcal{R}^{n,m,\lambda}\}_\lambda$ *with* $\mathcal{R}^{n,m,\lambda} := \{\mathcal{R}_k^{n,m,\lambda} : \mathcal{H}_{\mathrm{in}} \to \mathcal{H}_{\mathrm{out}}\}_{k \in \mathcal{K}}$ *satisfying:*

- ***Statistical Pseudorandomness***. *For any* $\ell = \mathrm{poly}(\lambda)$, *and any* $|\phi\rangle \in \mathcal{S}(\mathcal{H}_{\mathrm{in}})$,

$$\mathrm{TD}\left(\mathop{\mathbb{E}}_{k \leftarrow \mathcal{K}} \left[|\phi_k\rangle\langle\phi_k|^{\otimes \ell} \right], \mathop{\mathbb{E}}_{|\psi\rangle \in \mu} \left[|\psi\rangle\langle\psi|^{\otimes \ell} \right] \right) = \mathrm{negl}(\lambda)\,,$$

  where $|\phi_k\rangle := \mathcal{R}_k^{n,m,\lambda} |\phi\rangle$ *and* μ *is the Haar measure on* $\mathcal{S}(\mathcal{H}_{\mathrm{out}})$.
- ***Uniformity***. $\mathcal{R}^{n,m}$ *can be uniformly computed in polynomial time. That is, there is a deterministic Turing machine that, on input* $(1^n, 1^m, 1^\lambda, 1^\kappa)$, *outputs a quantum circuit* Q *in* $\mathrm{poly}(n, m, \lambda, \kappa)$ *time such that for all* $k \in \mathcal{K}$ *and* $|\phi\rangle \in \mathcal{S}(\mathcal{H}_{\mathrm{in}})$

$$Q\,|k\rangle|\phi\rangle = |k\rangle|\phi_k\rangle\,,$$

  where $|\phi_k\rangle := \mathcal{R}_k^{n,m,\lambda}|\phi\rangle$.

Table 1. A collection of quantum random and pseudorandom objects.

Random	Pseudorandom	Main property
Haar unitary	PRU $\{U_k\}$	$\{U_k\} \approx_c$ Haar unitary
RSS	PRSS $\{\mathcal{R}_k\}$	$\forall \left\vert\phi\right\rangle, \{\mathcal{R}_k \left\vert\phi\right\rangle\} \approx$ Haar state (trace distance or comp. indist.)
Haar state	PRSG $\{\mathcal{R}_k\}$	for some *fixed* $\left\vert\phi\right\rangle$ (e.g., $\left\vert 0\right\rangle$) $\{\mathcal{R}_k \left\vert\phi\right\rangle\} \approx_c$ Haar state

3.1 Properties of Pseudorandom State Scramblers

We discuss basic characteristics of the new primitives, as well as their relationships with pseudorandom state generators and their siblings.

Unitary to Isometry. It is sufficient to construct PRSSs from $\mathcal{H}$ to $\mathcal{H}$, since we can construct PRSSs from $\mathcal{H}_1$ to $\mathcal{H}_2$ ($n < m$) in the following way. Let $\mathcal{R}^{m,m} := \{\mathcal{R}^{m,m,\lambda}\}_\lambda$ be a PRSS with $\mathcal{R}^{m,m,\lambda} := \{\mathcal{R}_k^{m,m,\lambda} : \mathcal{H}_2 \to \mathcal{H}_2\}$. For all $\lambda \in \mathbb{N}$ and $k \in \mathcal{K}$, we define $\mathcal{R}_k^{n,m,\lambda} = \mathcal{R}_k^{m,m,\lambda}\left(\mathbb{1} \otimes \left\vert 0\right\rangle^{\otimes(m-n)}\right)$ where $\mathbb{1}$ is the identity of $\mathcal{H}_1$. It is not hard to verify that $\mathcal{R}^{n,m}$ is a PRSS from $\mathcal{H}_1$ to $\mathcal{H}_2$. We may write $\mathcal{R}^m$ instead of $\mathcal{R}^{m,m}$ when $m = n$.

Connections with Existing PRS *Variants.* Several definitions of quantum pseudorandomness on states with slight variations have been proposed and constructed since the regular PRS has been introduced. Brakerski and Shmueli [10] introduced scalable pseudorandom states (scalable PRSs) to eliminate the dependence between the state size and the security parameter. This modification aids in assuring the security when the state size n is much smaller than the security parameter λ. Ananth, Qian and Yuen [5] introduced pseudorandom function-like states (PRFSs), which extend PRSs by augmenting with classical inputs alongside the secret key. Although the security is initially based on pre-selected classical queries to the PRFS generator, the subsequent work [4] relaxes this to allow adversaries making adaptive (classical or quantum) queries resulting in three levels of security. The following theorem states that PRSSs subsume the original PRSs and those variants. The proof is deferred to the full version of this paper.

Theorem 2. PRSG*s, scalable* PRSG*s, and* PRFSG*s can be constructed via invoking* PRSS*s in a black-box manner.*

Oracle Separation from OWFs. According to [34, Theorem 2], PRUs exist relative to a quantum oracle $\mathcal{O}$, even when $\mathsf{BQP}^{\mathcal{O}} = \mathsf{QMA}^{\mathcal{O}}$, indicating the non-existence of one-way functions. Since PRUs imply PRSSs, we obtain the same oracle separation result for PRSSs.

Theorem 3. *There exists a quantum oracle $\mathcal{O}$ relative to which* PRSS*s exist, but* $\mathsf{BQP}^{\mathcal{O}} = \mathsf{QMA}^{\mathcal{O}}$.

4 Parallel Kac's Walk

In this section, we design a *parallel version* of the standard Kac's walk on $\mathcal{S}^n_{\mathbb{R}}$ [42] and demonstrate that it mixes exponentially faster with respect to the metrics of our interest. We assume $n = 2m$ for some $m \in \mathbb{N}$ throughout this section.

4.1 Parallel Kac's Walk on Real Space

Before introducing our parallel Kac's walk, we first review the standard one. The standard Kac's walk on vectors within a real Hilbert space is a Markov process. At each discrete time t, we randomly select two coordinates (i, j) of the vector, and then apply a two-dimensional rotation to the corresponding subvector with an angle θ drawn randomly and uniformly. After a predetermined number of steps, the Markov chain converges to a Haar distribution over the unit sphere. It is proved in [42] that the mixing time of Kac's walk on $\mathcal{S}^n_{\mathbb{R}}$ with respect to the total variation distance is $\Theta(n \log n)$. The formal definition of Kac's walk is given below.

Definition 6. *Kac's walk on $\mathcal{S}^n_{\mathbb{R}}$ is a discrete-time Markov chain $\{X_t \in \mathcal{S}^n_{\mathbb{R}}\}_{t\geq 0}$. At each time t, two coordinates $i^{(t)}, j^{(t)} \in [n]$ and an angle $\theta^{(t)} \in [0, 2\pi)$ are chosen uniformly at random. X_{t+1} is obtained by the following update rules:*

$$\begin{pmatrix} X_{t+1}[i^{(t)}] \\ X_{t+1}[j^{(t)}] \end{pmatrix} = \begin{bmatrix} \cos(\theta^{(t)}) & -\sin(\theta^{(t)}) \\ \sin(\theta^{(t)}) & \cos(\theta^{(t)}) \end{bmatrix} \begin{pmatrix} X_t[i^{(t)}] \\ X_t[j^{(t)}] \end{pmatrix},$$

$$X_{t+1}[k] = X_t[k] \quad \text{for} \quad k \notin \left\{ i^{(t)}, j^{(t)} \right\}.$$

We denote the Kac's walk as $G : [n] \times [n] \times [0, 2\pi) \times \mathcal{S}^n_{\mathbb{R}} \to \mathcal{S}^n_{\mathbb{R}}$ such that

$$X_{t+1} = G\left(i^{(t)}, j^{(t)}, \theta^{(t)}, X_t\right). \tag{1}$$

In our parallel Kac's walk, instead of randomly rotating one subvector, we simultaneously rotate m subvectors. Here we give its formal definition.

Definition 7. *The parallel Kac's walk is a discrete-time Markov chain $\{X_t \in \mathcal{S}^n_{\mathbb{R}}\}_{t\geq 0}$. At each step t, the parallel Kac's walk first selects a random perfect matching of the set $\{1, \ldots, n\}$, denoted by*

$$P_t = \left\{ \left(i_1^{(t)}, j_1^{(t)}\right), \ldots, \left(i_m^{(t)}, j_m^{(t)}\right) \right\},$$

where $\bigcup_{k=1}^{m} \left\{ i_k^{(t)}, j_k^{(t)} \right\} = \{1, \ldots, n\}$. Then m independent angles $\theta_1^{(t)}, \ldots, \theta_m^{(t)} \in [0, 2\pi)$ are chosen uniformly at random. For every pair $\left(i_k^{(t)}, j_k^{(t)}\right)$ in P_t, it sets

$$\begin{pmatrix} X_{t+1}[i_k^{(t)}] \\ X_{t+1}[j_k^{(t)}] \end{pmatrix} = \begin{bmatrix} \cos(\theta_k^{(t)}) & -\sin(\theta_k^{(t)}) \\ \sin(\theta_k^{(t)}) & \cos(\theta_k^{(t)}) \end{bmatrix} \begin{pmatrix} X_t[i_k^{(t)}] \\ X_t[j_k^{(t)}] \end{pmatrix}.$$

Let $F : ([n] \times [n])^m \times [0, 2\pi)^m \times \mathcal{S}_{\mathbb{R}}^n \to \mathcal{S}_{\mathbb{R}}^n$ *denote the map associated with the above random walk such that*

$$X_{t+1} = F\left(P_t, \theta_1^{(t)}, \ldots, \theta_m^{(t)}, X_t\right). \tag{2}$$

In one step of the parallel Kac's walk, we obtain m distinct coordinate pairs by randomly sampling a perfect matching P_t of set $[n]$. For each pair, a rotation angle is selected independently and uniformly at random. Recall the notation in Definition 6. Let $X_{t,1} = X_t$ and $X_{t,k+1} = G\left(i_k^{(t)}, j_k^{(t)}, \theta_k^{(t)}, X_{t,k}\right)$ for $1 \le k \le m$. It is evident that

$$X_{t,m+1} = X_{t+1} = F\left(P_t, \theta_1^{(t)}, \ldots, \theta_m^{(t)}, X_t\right).$$

We can observe that taking one step of the parallel Kac's walk can be viewed as taking $m = n/2$ steps in the original Kac's walk when there are *no collisions* in the pairing step. All the subvectors being rotated in a single step of the parallel Kac's walk are distinct, and thus not independent. Consequently, the results for the original Kac's walk cannot be directly applied. Fortunately, by enhancing the coupling technique for analyzing the mixing time of the standard Kac's walk, we are able to prove that the parallel Kac's walk rapidly mixes in time $O(\log n)$ with respect to two different metrics: (1) the Wasserstein 1-distance; and (2) the total variation distance.

In the context of the Wasserstein 1-distance, after walking T steps, the difference between the output distribution of a parallel Kac's walk and the normalized Haar measure decays exponentially as T grows, which leads to a $O(\log n)$ mixing time. Formally,

Theorem 4. *Let* $\{X_t \in \mathcal{S}_{\mathbb{R}}^n\}_{t \ge 0}$ *be a Markov chain that evolves according to the parallel Kac's walk. Then, for sufficiently large* n, $c > 0$, *and* $T = 10(c+1)\log n$,

$$\sup_{X_0 \in \mathcal{S}_{\mathbb{R}}^n} W_1(\mathcal{L}(X_T), \mu) \le \frac{1}{2^{c \log n}},$$

where μ *is the normalized Haar measure on* $\mathcal{S}_{\mathbb{R}}^n$.

Furthermore, we get a stronger result regarding the total variation distance:

Theorem 5. *Let* $\{X_t \in \mathcal{S}_{\mathbb{R}}^n\}_{t \ge 0}$ *be a Markov chain that evolves according to the parallel Kac's walk. Then, for sufficiently large* n, $c > 515$ *and* $T = c \log n$,

$$\sup_{X_0 \in \mathcal{S}_{\mathbb{R}}^n} \|\mathcal{L}(X_T) - \mu\|_{\mathrm{TV}} \le \frac{1}{2^{(c/515-1)\log n - 1}},$$

where μ *is the normalized Haar measure on* $\mathcal{S}_{\mathbb{R}}^n$.

Notably, while the Wasserstein 1-distance is a weaker metric compared to the total variation distance, Theorem 4 provides an adequate foundation for constructing a PRSS. Additionally, the analysis of Theorem 5 further reveals a *dispersing property* of our construction of RSS. The remainder of this section is devoted to proving Theorem 4. The proof for Theorem 5, is deferred to the full version of this paper.

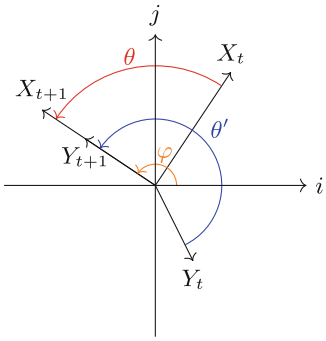

Fig. 1. Transformation of subcoordinates $X_t[i,j]$ and $Y_t[i,j]$

4.2 The Proportional Coupling

Our technique for proving the mixing time in Theorem 4 accommodates the *proportional coupling* [42] that sufficiently reduces the distance between two copies of Kac's walk. At each time t in the proportional coupling (illustrated in Fig. 1), an angle θ is chosen uniformly at random from $[0, 2\pi)$ for rotating the subvector $(X_t[i], X_t[j])$, where indices i and j are picked as in Definition 6. The angle θ' is specifically selected for $(Y_t[i], Y_t[j])$ to make it collinear with $(X_t[i], X_t[j])$, i.e., they share the same argument φ. Taking into account the marginal distribution, both θ and θ' are drawn from the uniform distribution over the interval $[0, 2\pi)$, validating the proportional coupling for two Kac's walks.

Following a similar idea, we define the proportional coupling of two copies of the parallel Kac's walk, which couples each pair of indices from the randomly sampled perfect matching using the proportional coupling.

Definition 8 (Proportional Coupling for the Parallel Kac's Walk). *We define a coupling of two copies $\{X_t\}_{t\geq 0}, \{Y_t\}_{t\geq 0}$ of the parallel Kac's walk in the following way: Fix X_t, $Y_t \in \mathcal{S}^n_{\mathbb{R}}$.*

1. *Choose a perfect matching $P_t = \left\{\left(i_1^{(t)}, j_1^{(t)}\right), \ldots, \left(i_m^{(t)}, j_m^{(t)}\right)\right\}$ and m angles $\theta_1^{(t)}, \ldots, \theta_m^{(t)} \in [0, 2\pi)$ uniformly at random, and set*

$$X_{t+1} = F\left(P_t, \theta_1^{(t)}, \ldots, \theta_m^{(t)}, X_t\right).$$

2. *Sample m angles $\theta'^{(t)}_1, \ldots, \theta'^{(t)}_m$ in the following manner: for every $1 \leq k \leq m$, (a) choose $\varphi_k \in [0, 2\pi)$ uniformly at random among all angles that satisfy*

$$X_{t+1}[i_k^{(t)}] = \sqrt{X_t[i_k^{(t)}]^2 + X_t[j_k^{(t)}]^2}\cos(\varphi_k),$$
$$X_{t+1}[j_k^{(t)}] = \sqrt{X_t[i_k^{(t)}]^2 + X_t[j_k^{(t)}]^2}\sin(\varphi_k),$$

(b) and then choose $\theta'^{(t)}_k \in [0, 2\pi)$ *uniformly among the angles that satisfy*

$$\cos(\theta'^{(t)}_k) \cdot Y_t[i_k^{(t)}] - \sin(\theta'^{(t)}_k) \cdot Y_t[j_k^{(t)}] = \sqrt{Y_t[i_k^{(t)}]^2 + Y_t[j_k^{(t)}]^2}\cos(\varphi_k),$$
$$\sin(\theta'^{(t)}_k) \cdot Y_t[i_k^{(t)}] + \cos(\theta'^{(t)}_k) \cdot Y_t[j_k^{(t)}] = \sqrt{Y_t[i_k^{(t)}]^2 + Y_t[j_k^{(t)}]^2}\sin(\varphi_k).$$

And set $Y_{t+1} = F\left(P_t, \theta'^{(t)}_1, \ldots, \theta'^{(t)}_m, Y_t\right)$.

In this coupling scheme, we enforce X_t and Y_t to employ an identical random matching (P_t in step 1) to generate all the m pairs of coordinates. And then we sample m rotation angles for X_t and obtain X_{t+1} by rotating the m coordinate pairs by their corresponding angles. Next, in step 2, we determine the rotation angle for each coordinate pair of Y_t. For the k-th pair, our objective is to select a suitable angle $\theta'^{(t)}_k$ such that the two-dimensional subvector $(Y_{t+1}[i_k^{(t)}], Y_{t+1}[j_k^{(t)}])$ aligns collinearly with $(X_{t+1}[i_k^{(t)}], X_{t+1}[j_k^{(t)}])$. To achieve this, we ensure that $(Y_{t+1}[i_k^{(t)}], Y_{t+1}[j_k^{(t)}])$ shares the same argument φ_k as $(X_{t+1}[i_k^{(t)}], X_{t+1}[j_k^{(t)}])$. Typically, the values of angles φ_k and $\theta'^{(t)}_k$ are uniquely determined. However, in the scenario where either $(X_t[i_k^{(t)}], X_t[j_k^{(t)}])$ or $(Y_t[i_k^{(t)}], Y_t[j_k^{(t)}])$ equals the zero vector, all angles satisfy the required conditions. In such cases, we resort to uniform random selection for determining the angles.

Remark 1. This coupling forces $X_{t+1}[i]Y_{t+1}[i] \geq 0$ for all $i \in [n]$ since the signs are determined by the same arguments.

In each step of our coupling scheme, a quarter of the distance between vectors X_t and Y_t is reduced, which is formally shown in

Lemma 3. *Let* $X_0, Y_0 \in \mathcal{S}^n_{\mathbb{R}}$. *For* $t \geq 0$, *we couple* (X_{t+1}, Y_{t+1}) *conditioned on* (X_t, Y_t) *according to the proportional coupling defined in Definition 8. We define*

$$A_t[i] = X_t[i]^2, \quad B_t[i] = Y_t[i]^2.$$

Then for any $l \in \mathbb{N}$, *we have*

$$\mathbb{E}\left[\sum_{i=1}^{n} \left(A_l[i] - B_l[i]\right)^2\right] \leq 2 \cdot \left(1 - \frac{1}{4}\right)^l.$$

Proof. Fix $X_t, Y_t \in \mathcal{S}^n_{\mathbb{R}}$. Let (X_{t+1}, Y_{t+1}) obtained from (X_t, Y_t) by applying the coupling defined in Definition 8. Recall that $n = 2m$. Let $N = \frac{n!}{2^m m!}$ be the number of perfect matchings for $[n]$. To keep the notations short, the perfect matching $\left\{\left(i_1^{(t)}, j_1^{(t)}\right), \ldots, \left(i_m^{(t)}, j_m^{(t)}\right)\right\}$ at step t is denoted by $\left(\overrightarrow{i^{(t)}}, \overrightarrow{j^{(t)}}\right)$.

We have

$$\begin{aligned}&\mathbb{E}\left[\sum_{i=1}^{n}\left(A_{t+1}[i]-B_{t+1}[i]\right)^2\right]\\&=\frac{1}{N}\sum_{\left(\overrightarrow{i^{(t)}},\overrightarrow{j^{(t)}}\right)}\underbrace{\mathbb{E}\left[\sum_{i=1}^{n}\left(A_{t+1}[i]-B_{t+1}[i]\right)^2\middle|P_t=\left(\overrightarrow{i^{(t)}},\overrightarrow{j^{(t)}}\right)\right]}_{(\star)}.\end{aligned}\tag{3}$$

By the definition of the parallel Kac's walk, we have

$$\begin{aligned}(\star)=&\sum_{k=1}^{m}\mathbb{E}\left[\left(\left(A_t[i_k^{(t)}]+A_t[j_k^{(t)}]\right)\cos(\varphi_k)^2-\left(B_t[i_k^{(t)}]+B_t[j_k^{(t)}]\right)\cos(\varphi_k)^2\right)^2\right]\\&+\sum_{k=1}^{m}\mathbb{E}\left[\left(\left(A_t[i_k^{(t)}]+A_t[j_k^{(t)}]\right)\sin(\varphi_k)^2-\left(B_t[i_k^{(t)}]+B_t[j_k^{(t)}]\right)\sin(\varphi_k)^2\right)^2\right]\\=&\frac{3}{4}\sum_{k=1}^{m}\left(\left(A_t[i_k^{(t)}]+A_t[j_k^{(t)}]\right)-\left(B_t[i_k^{(t)}]+B_t[j_k^{(t)}]\right)\right)^2\\=&\underbrace{\frac{3}{4}\sum_{k=1}^{m}\left(\left(A_t[i_k^{(t)}]-B_t[i_k^{(t)}]\right)^2+\left(A_t[j_k^{(t)}]-B_t[j_k^{(t)}]\right)^2\right)}_{(\star\star)}\\&+\underbrace{\frac{3}{4}\sum_{k=1}^{m}2\left(A_t[i_k^{(t)}]-B_t[i_k^{(t)}]\right)\left(A_t[j_k^{(t)}]-B_t[j_k^{(t)}]\right)}_{(\star\star\star)},\end{aligned}\tag{4}$$

where the second equality is by $\mathbb{E}\left[\cos(\varphi_k)^4\right]=\mathbb{E}\left[\sin(\varphi_k)^4\right]=3/8$. As $\left\{\left(i_1^{(t)},j_1^{(t)}\right),\ldots,\left(i_m^{(t)},j_m^{(t)}\right)\right\}$ is a perfect matching, we have

$$(\star\star)=\frac{3}{4}\sum_{i=1}^{n}\left(A_t[i]-B_t[i]\right)^2.\tag{5}$$

Combining Eqs. (3), (4), and (5), we obtain

$$\mathbb{E}\left[\sum_{i=1}^{n}\left(A_{t+1}[i]-B_{t+1}[i]\right)^2\right]=\frac{3}{4}\sum_{i=1}^{n}\left(A_t[i]-B_t[i]\right)^2+\underbrace{\frac{1}{N}\sum_{\left(\overrightarrow{i^{(t)}},\overrightarrow{j^{(t)}}\right)}(\star\star\star)}_{(4\star)}.\tag{6}$$

For the last term,

$$
\begin{aligned}
(4\star) =& \frac{3}{2N} \sum_{\left(\overrightarrow{i^{(t)}}, \overrightarrow{j^{(t)}}\right)} \sum_{k=1}^{m} \left(A_t[i_k^{(t)}] - B_t[i_k^{(t)}]\right) \left(A_t[j_k^{(t)}] - B_t[j_k^{(t)}]\right) \\
=& \frac{3}{2N} \cdot \frac{(n-2)!}{2^{m-1}(m-1)!} \sum_{i<j} \left(A_t[i] - B_t[i]\right) \left(A_t[j] - B_t[j]\right) \\
=& \frac{3 \cdot m}{2n(n-1)} \left(\left(\sum_{i=1}^{n} \left(A_t[i] - B_t[i]\right) \right)^2 - \sum_{i=1}^{n} \left(A_t[i] - B_t[i]\right)^2 \right) \\
=& -\frac{3}{4(n-1)} \sum_{i=1}^{n} \left(A_t[i] - B_t[i]\right)^2 . \qquad (7)
\end{aligned}
$$

Combining Eqs. (6) and (7), we have

$$
\begin{aligned}
\mathbb{E}\left[\sum_{i=1}^{n} \left(A_l[i] - B_l[i]\right)^2 \right] &= \mathbb{E}\left[\mathbb{E}\left[\sum_{i=1}^{n} \left(A_l[i] - B_l[i]\right)^2 \middle| X_{l-1}, Y_{l-1} \right] \right] \\
&\leq \frac{3}{4} \mathbb{E}\left[\sum_{i=1}^{n} \left(A_{l-1}[i] - B_{l-1}[i]\right)^2 \right] \\
&\leq \left(\frac{3}{4}\right)^l \sum_{i=1}^{n} \left(A_0[i] - B_0[i]\right)^2 \leq 2 \cdot \left(\frac{3}{4}\right)^l .
\end{aligned}
$$

□

4.3 Proof of the Mixing Time

Proof of Theorem 4. Let $T = 10(c+1)\log n$ for $c > 0$. We couple two copies $\{X_t\}_{t\geq 0}$ and $\{Y_t\}_{t\geq 0}$ of the parallel Kac's walk with starting points $X_0 = x \in \mathcal{S}_{\mathbb{R}}^n$ and $Y_0 \sim \mu$, by applying the proportional coupling. We have

$$
W_1(\mathcal{L}(X_T), \mu) = W_1(\mathcal{L}(X_T), \mathcal{L}(Y_T)) \leq \mathbb{E}\left[\|X_T - Y_T\|_2\right] \leq \left(\mathbb{E}\left[\|X_T - Y_T\|_2^4\right]\right)^{1/4} .
$$

Then by Cauchy-Schwarz inequality, we have

$$
W_1(\mathcal{L}(X_T), \mu) \leq \left(n\, \mathbb{E}\left[\|X_T - Y_T\|_4^4\right]\right)^{1/4} . \qquad (8)
$$

Note that the proportional coupling forces $X_T[i]Y_T[i] \geq 0$ for all $i \in [n]$. Therefore, for all $i \in [n]$

$$
|X_T[i] - Y_T[i]| \leq |X_T[i] + Y_T[i]| .
$$

This gives us

$$\|X_T - Y_T\|_4^4 = \sum_{i=1}^{n} \left(X_T[i] - Y_T[i]\right)^4 \leq \sum_{i=1}^{n} \left(X_T[i]^2 - Y_T[i]^2\right)^2. \tag{9}$$

Combining Eqs. (8) and (9), we have

$$\begin{aligned} W_1(\mathcal{L}(X_T), \mu) &\leq \left(n\, \mathbb{E}\left[\sum_{i=1}^{n} \left(X_T[i]^2 - Y_T[i]^2\right)^2 \right] \right)^{1/4} \\ \text{(Lemma 3)} &\leq \left(2n \left(\frac{3}{4}\right)^T \right)^{1/4} \leq \frac{1}{2^{c \log n}}. \end{aligned}$$

□

5 Constructions of RSSs and PRSSs

In this section, we present a family of circuits that implements RSSs, specifically realizing the parallel Kac's walk on the real unit sphere. To obtain circuits for PRSSs, one can simply replace the random primitives with their post-quantum secured pseudorandom counterparts. Lastly, we discuss the dispersing property.

5.1 Simulating One Single Step

We begin by constructing a unitary gate that simulates a single step of the parallel Kac's walk on $\mathcal{S}_{\mathbb{R}}^{2^n}$. In every step, we denote the corresponding permutation by $\sigma \in S_{2^n}$. And we use the function $f : \{0,1\}^{n-1} \to \{0,1\}^d$ to manage the precision of the rotation angle that was originally chosen from the interval $[0, 2\pi)$, where d is the parameter controlling the precision of the rotation angle. Specifically, for every σ and f, we define a unitary gate $K_{\sigma,f} = U_{\sigma^{-1}} W_f U_\sigma$, where

$$U_\sigma = \sum_{x \in \{0,1\}^n} |\sigma(x)\rangle\langle x|, \quad W_f = \sum_{y \in \{0,1\}^{n-1}} \begin{pmatrix} \cos(\theta_y) & -\sin(\theta_y) \\ \sin(\theta_y) & \cos(\theta_y) \end{pmatrix} \otimes |y\rangle\langle y|, \tag{10}$$

and $\theta_y = 2\pi \cdot \mathrm{val}(f(y))$ is the rotation angle for every subvector $\left(\sigma^{-1}(0y), \sigma^{-1}(1y)\right)$, $y \in \{0,1\}^{n-1}$. In Fig. 2, we show a quantum circuit that realizes $K_{\sigma,f}$.

The circuit consists of:

1. Permutation: a unitary U_σ which transforms $|x\rangle$ to $|\sigma(x)\rangle$ for any $x \in \{0,1\}^n$. This unitary can be implemented via making quires to oracles O_σ and $O_{\sigma^{-1}}$, and using n ancilla qubits: for any $x \in \{0,1\}^n$,

$$|x\rangle|0\rangle \xrightarrow{O_\sigma} |x\rangle|\sigma(x)\rangle \xrightarrow{SWAP} |\sigma(x)\rangle|x\rangle \xrightarrow{O_{\sigma^{-1}}} |\sigma(x)\rangle|0\rangle.$$

We omit this detail in the above figure for the sake of conciseness.

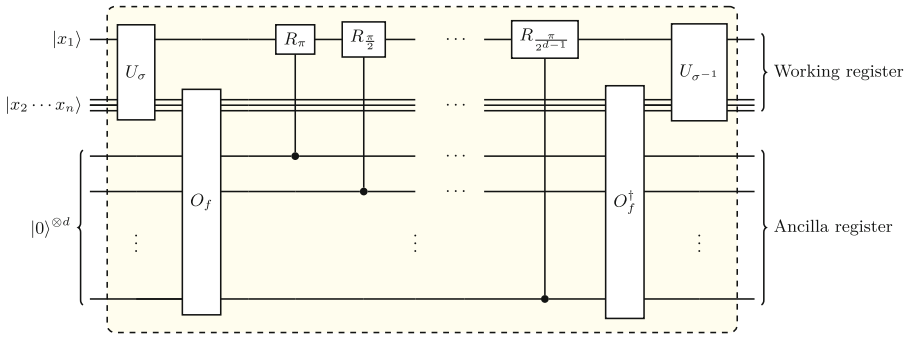

Fig. 2. Circuit diagram for the construction of the $K_{\sigma,f}$

2. Implementing rotation operator W_f:
   (a) an oracle O_f which queries $f(x_2, \ldots, x_n)$ and stores the d-bit result in the ancilla qubits.
   (b) d controlled-rotation gates. The i-th ancilla qubit controls $R_{\frac{\pi}{2^{i-1}}}$ gate acting on the first qubit, where the gate R_θ denotes the rotation transformation $\begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$.
   (c) an oracle O_f again for uncomputing the ancilla qubits.
3. Inverse permutation: a unitary $U_{\sigma^{-1}}$.

Remark. The gate $K_{\sigma,f}$ approximates one step of the parallel Kac's walk. It starts by partitioning the computational basis (indices) into 2^{n-1} pairs based on a selected permutation σ. For each pair $\left(\sigma^{-1}(0y), \sigma^{-1}(1y)\right)$ labeled by $y \in \{0,1\}^{n-1}$, the gate applies a rotation with an approximated angle θ_y indicated by f to the corresponding two dimensional subvector.

Stepwise State Evolution. To gain insight into the functionality of $K_{\sigma,f}$, we assume the initial state to be a pure state

$$|\varphi\rangle = \sum_{x\in\{0,1\}^n} p_x |x\rangle .$$

First, to pair up the indices by applying U_σ, the initial state is transformed into

$$\begin{aligned} \sum_{x\in\{0,1\}^n} p_x |\sigma(x)\rangle \otimes \left|0^d\right\rangle &= \sum_{x'\in\{0,1\}^n} p_{\sigma^{-1}(x')} |x'\rangle \otimes \left|0^d\right\rangle \\ &= \sum_{y\in\{0,1\}^{n-1}} \left(p_{\sigma^{-1}(0y)} |0\rangle + p_{\sigma^{-1}(1y)} |1\rangle\right) \otimes |y\rangle \otimes \left|0^d\right\rangle . \end{aligned}$$

Then, to rotate each subvector, the oracle O_f stores $f(y)$ in the ancilla register as control qubits, resulting in the state

$$\sum_{y\in\{0,1\}^{n-1}} \left(p_{\sigma^{-1}(0y)}\left|0\right\rangle + p_{\sigma^{-1}(1y)}\left|1\right\rangle\right) \otimes \left|y\right\rangle \otimes \left|f(y)\right\rangle.$$

Next, a series of controlled-rotation gates are applied to the first qubit, rotating it by an angle of $\theta_y = 2\pi \cdot \mathrm{val}(f(y))$. Therefore, we have the following state:

$$\sum_{y\in\{0,1\}^{n-1}} \left(p'_{\sigma^{-1}(0y)}\left|0\right\rangle + p'_{\sigma^{-1}(1y)}\left|1\right\rangle\right) \otimes \left|y\right\rangle \otimes \left|f(y)\right\rangle$$

where

$$\begin{aligned} p'_{\sigma^{-1}(0y)} &= \cos(\theta_y)\cdot p_{\sigma^{-1}(0y)} - \sin(\theta_y)\cdot p_{\sigma^{-1}(1y)},\\ p'_{\sigma^{-1}(1y)} &= \sin(\theta_y)\cdot p_{\sigma^{-1}(1y)} + \cos(\theta_y)\cdot p_{\sigma^{-1}(1y)}. \end{aligned}$$

After reverting the ancilla qubits and applying the inverse permutation, we obtain the output state

$$\sum_{y\in\{0,1\}^{n-1}} \left(p'_{\sigma^{-1}(0y)}\left|\sigma^{-1}(0y)\right\rangle + p'_{\sigma^{-1}(1y)}\left|\sigma^{-1}(1y)\right\rangle\right) \otimes \left|0^d\right\rangle = \sum_{x\in\{0,1\}^n} p'_x \left|x\right\rangle \otimes \left|0^d\right\rangle.$$

5.2 Constructing RSS

We first define an ensemble RSG^n of unitary operators that represents applying $K_{\sigma,f}$ for T-step with i.i.d. random selections of permutations and functions. Then, we prove that such an ensemble forms an RSS.

Definition 9. *Let $n, T, d \in \mathbb{N}$, and $\mathcal{H}$ be a real Hilbert space with dimension 2^n. An ensemble of unitary operators $\mathsf{RSG}^n :- \left\{\mathsf{RSG}^{n,\lambda}\right\}_\lambda$ with*

$$\mathsf{RSG}^{n,\lambda} :- \left\{\mathsf{RSG}^{n,\lambda}_{(\sigma_i)_{i=1}^T,(f_i)_{i=1}^T} : \mathcal{H}\to\mathcal{H}\right\}_{(\sigma_i)_{i=1}^T\subseteq S_{2^n},(f_i)_{i=1}^T\subseteq\{f:\{0,1\}^{n-1}\to\{0,1\}^d\}}$$

is define as

$$\mathsf{RSG}^{n,\lambda}_{(\sigma_i)_{i=1}^T,(f_i)_{i=1}^T} = K_{\sigma_T,f_T}\cdots K_{\sigma_2,f_2}K_{\sigma_1,f_1}$$

where $K_{\sigma,f} = U_{\sigma^{-1}}W_f U_\sigma$ is defined in (10).

Theorem 6. *Let $n\in\mathbb{N}$, $d = \log^2\lambda + \log^2 n$ and $T = 10(\lambda+1)n$. The ensemble of unitary operators RSG^n defined in Definition 9 is an RSS.*

To prove Theorem 6, we define a new ensemble of (infinitely many) unitary operators $\widetilde{\mathsf{RSG}}^n :- \left\{\widetilde{\mathsf{RSG}}^{n,\lambda}\right\}_\lambda$ with

$$\widetilde{\mathsf{RSG}}^{n,\lambda} :- \left\{\widetilde{\mathsf{RSG}}^{n,\lambda}_{(\sigma_i)_{i=1}^T,(\widetilde{f}_i)_{i=1}^T} : \mathcal{H}\to\mathcal{H}\right\}_{(\sigma_i)_{i=1}^T\subseteq S_{2^n},(\widetilde{f}_i)_{i=1}^T\subseteq\{f:\{0,1\}^{n-1}\to[0,1)\}}$$

and

$$\widetilde{\mathsf{RSG}}^{n,\lambda}_{(\sigma_i)_{i=1}^T,(\widetilde{f}_i)_{i=1}^T} = \widetilde{K}_{\sigma_T,\widetilde{f}_T} \cdots \widetilde{K}_{\sigma_2,\widetilde{f}_2} \widetilde{K}_{\sigma_1,\widetilde{f}_1}$$

where $\widetilde{K}_{\sigma,\widetilde{f}} = U_{\sigma^{-1}} \widetilde{W}_{\widetilde{f}} U_\sigma$ and $\widetilde{W}_{\widetilde{f}}$ is defined to be

$$\widetilde{W}_{\widetilde{f}} = \sum_{y \in \{0,1\}^{n-1}} \begin{pmatrix} \cos\left(\widetilde{\theta}_y\right) & -\sin\left(\widetilde{\theta}_y\right) \\ \sin\left(\widetilde{\theta}_y\right) & \cos\left(\widetilde{\theta}_y\right) \end{pmatrix} \otimes |y\rangle\langle y| \,, \tag{11}$$

in which $\widetilde{\theta}_y = 2\pi \cdot \widetilde{f}(y)$ for $y \in \{0,1\}^{n-1}$.

RSG^n and $\widetilde{\mathsf{RSG}}^n$ differ in the way the angles are chosen. In RSG^n, the angles are selected from the discrete set $\left\{2\pi \cdot \frac{i}{2^d} : i \in \left\{0, 1, \ldots, 2^d - 1\right\}\right\}$, while in $\widetilde{\mathsf{RSG}}^n$, the angles are chosen from the interval $[0, 2\pi)$. For uniformly random σ and $\widetilde{f}$, applying gate $\widetilde{K}_{\sigma,\widetilde{f}}$ results in the selection of a random matching on the computational basis, with each pair in the matching being rotated by a random angle in $[0, 2\pi)$ determined by the corresponding value of $\widetilde{f}$. This is exactly one step of parallel Kac's walk described in Sect. 4. $\widetilde{\mathsf{RSG}}^n$ serves as an intermediate scrambler in the proof of Theorem 6. To analyse the difference between RSG^n and $\widetilde{\mathsf{RSG}}^n$, we need the following lemma.

Lemma 4. *Let $\sigma \in S_{2^n}$ and $\widetilde{f} : \{0,1\}^{n-1} \to [0,1)$. Let f be the function satisfying for any $y \in \{0,1\}^{n-1}$, $f(y)$ is the d digits after the binary point in $\widetilde{f}(y)$. Then*

$$\left\| K_{\sigma,f} - \widetilde{K}_{\sigma,\widetilde{f}} \right\|_\infty \leq 2^{1-d}\pi,$$

where $K_{\sigma,f} = U_{\sigma^{-1}} W_f U_\sigma$ is defined in (10) and $\widetilde{K}_{\sigma,\widetilde{f}} = U_{\sigma^{-1}} \widetilde{W}_{\widetilde{f}} U_\sigma$ is defined in (11).

Proof.

$$\begin{aligned}
\left\| K_{\sigma,f} - \widetilde{K}_{\sigma,\widetilde{f}} \right\|_\infty &= \left\| U_{\sigma^{-1}} \left(W_f - \widetilde{W}_{\widetilde{f}} \right) U_\sigma \right\|_\infty \\
&= \left\| \sum_{y \in \{0,1\}^{n-1}} \begin{pmatrix} \cos\theta_y - \cos\widetilde{\theta}_y & -\left(\sin\theta_y - \sin\widetilde{\theta}_y\right) \\ \sin\theta_y - \sin\widetilde{\theta}_y & \cos\theta_y - \cos\widetilde{\theta}_y \end{pmatrix} \otimes |y\rangle\langle y| \right\|_\infty \\
&= \max_{y \in \{0,1\}^{n-1}} \left\{ 2 \left| \sin \frac{\theta_y - \widetilde{\theta}_y}{2} \right| \left\| \begin{pmatrix} -\sin\frac{\theta_y + \widetilde{\theta}_y}{2} & -\cos\frac{\theta_y + \widetilde{\theta}_y}{2} \\ \cos\frac{\theta_y + \widetilde{\theta}_y}{2} & -\sin\frac{\theta_y + \widetilde{\theta}_y}{2} \end{pmatrix} \right\|_\infty \right\} \\
&\leq \max_{y \in \{0,1\}^{n-1}} \left\{ \left| \theta_y - \widetilde{\theta}_y \right| \right\} \leq 2^{1-d}\pi.
\end{aligned}$$

□

Proof of Theorem 6. It is easy to see that the uniformity condition is satisfied. Let κ denote the key length. Quantum circuit $\mathsf{RSG}^{n,\lambda}$ applies $\mathsf{RSG}^{n,\lambda}_{(\sigma_i)_{i=1}^T,(f_i)_{i=1}^T}$ after reading $(\sigma_i)_{i=1}^T$ and $(f_i)_{i=1}^T$. To implement $\mathsf{RSG}^{n,\lambda}_{(\sigma_i)_{i=1}^T,(f_i)_{i=1}^T}$, we need to realize

each of the $T = 10(\lambda+1)n$ unitary gates K. Since each gate K can be implemented in $\mathrm{poly}(n,\lambda,\kappa)$ time, the total construction time for $\mathsf{RSG}^{n,\lambda}_{(\sigma_i)_{i=1}^T,(f_i)_{i=1}^T}$ is also $\mathrm{poly}(n,\lambda,\kappa)$.

Thus, it suffices to prove the requirement of *Statistical Pseudorandomness* is satisfied. Fix $|\eta\rangle \in \mathcal{S}(\mathcal{H})$. Define three distributions:

- ν be the distribution of $\mathsf{RSG}^{n,\lambda}_{(\sigma_i)_{i=1}^T,(f_i)_{i=1}^T}|\eta\rangle$ with independent and uniformly random permutations $(\sigma_i)_{i=1}^T \subseteq S_{2^n}$ and random functions $(f_i)_{i=1}^T \subseteq \{f : \{0,1\}^{n-1} \to \{0,1\}^d\}$.
- $\widetilde{\nu}$ be the distribution of $\widetilde{\mathsf{RSG}}^{n,\lambda}_{(\sigma_i)_{i=1}^T,(\widetilde{f}_i)_{i=1}^T}|\eta\rangle$ with independent and uniformly random permutations $(\sigma_i)_{i=1}^T \subseteq S_{2^n}$, and random functions $(\widetilde{f}_i)_{i=1}^T \subseteq \{f : \{0,1\}^{n-1} \to [0,1)\}$.
- μ be the Haar measure on $\mathcal{S}^{2^n}_{\mathbb{R}}$.

We first prove that the trace distance between ν and $\widetilde{\nu}$ is negligible. To this end, we construct a coupling γ_0 of ν and $\widetilde{\nu}$ by using the same permutation σ_t and letting f_t be the function satisfying $f_t(y)$ is the d digits after the binary point in $\widetilde{f}_t(y)$ for all $y \in \{0,1\}^{n-1}$. Therefore, for any $(|\phi\rangle, |\varphi\rangle) \sim \gamma_0$, we have

$$\begin{aligned}
\||\phi\rangle - |\varphi\rangle\|_2 &= \left\| \widetilde{\mathsf{RSG}}^{n,\lambda}_{(\sigma_i)_{i=1}^T,(\widetilde{f}_i)_{i=1}^T}|\eta\rangle - \mathsf{RSG}^{n,\lambda}_{(\sigma_i)_{i=1}^T,(f_i)_{i=1}^T}|\eta\rangle \right\|_2 \\
&\le \left\| \widetilde{\mathsf{RSG}}^{n,\lambda}_{(\sigma_i)_{i=1}^T,(\widetilde{f}_i)_{i=1}^T} - \mathsf{RSG}^{n,\lambda}_{(\sigma_i)_{i=1}^T,(f_i)_{i=1}^T} \right\|_\infty \\
&\le 2^{1-d}\pi T = \frac{20\pi(\lambda+1)n}{\lambda^{\log\lambda}\cdot n^{\log n}},
\end{aligned}$$

where the last inequality is from Fact 1 and Lemma 4. Thus, for any $l \in \mathrm{poly}(\lambda, n)$

$$\begin{aligned}
&\left\| \mathop{\mathbb{E}}_{|\phi\rangle\sim\nu}\left[(|\phi\rangle\langle\phi|)^{\otimes l}\right] - \mathop{\mathbb{E}}_{|\varphi\rangle\sim\widetilde{\nu}}\left[(|\varphi\rangle\langle\varphi|)^{\otimes l}\right] \right\|_1 \\
\le\ & \mathop{\mathbb{E}}_{(|\phi\rangle,|\varphi\rangle)\sim\gamma_0}\left[\left\|(|\phi\rangle\langle\phi|)^{\otimes l} - (|\varphi\rangle\langle\varphi|)^{\otimes l}\right\|_1\right] \\
\le\ & l \mathop{\mathbb{E}}_{(|\phi\rangle,|\varphi\rangle)\sim\gamma_0}\left[\||\phi\rangle\langle\phi| - |\varphi\rangle\langle\varphi|\|_1\right] \\
\le\ & l\left(\mathop{\mathbb{E}}_{(|\phi\rangle,|\varphi\rangle)\sim\gamma_0}\left[\||\phi\rangle\left(\langle\phi| - \langle\varphi|\right)\|_1\right] + \mathop{\mathbb{E}}_{(|\phi\rangle,|\varphi\rangle)\sim\gamma_0}\left[\|\left(|\phi\rangle - |\varphi\rangle\right)\langle\varphi|\|_1\right]\right) \\
\le\ & 2l \mathop{\mathbb{E}}_{(|\phi\rangle,|\varphi\rangle)\sim\gamma_0}\left[\||\phi\rangle - |\varphi\rangle\|_2\right] \le \frac{40\pi(\lambda+1)nl}{\lambda^{\log\lambda}\cdot n^{\log n}}.
\end{aligned} \tag{12}$$

As for the trace distance between $\widetilde{\nu}$ and μ, note that $\widetilde{\nu}$ is the output distribution of T-step parallel Kac's walk. Thus by Theorem 4, we have

$$W_1(\widetilde{\nu},\mu) \le \frac{1}{2^{\lambda n}}.$$

So there exists a coupling of $\widetilde{v}$ and μ, denoted by γ_1, that achieves

$$\mathop{\mathbb{E}}_{(|\varphi\rangle,|\psi\rangle)\sim\gamma_1}[\||\varphi\rangle - |\psi\rangle\|_2] \leq \frac{3}{2^{\lambda n}}.$$

Therefore, similar to Eq. (12), we have for any $l \in \mathrm{poly}(\lambda, n)$

$$\left\| \mathop{\mathbb{E}}_{|\varphi\rangle\sim\widetilde{\nu}} \left[(|\varphi\rangle\langle\varphi|)^{\otimes l} \right] - \mathop{\mathbb{E}}_{|\psi\rangle\sim\mu} \left[(|\psi\rangle\langle\psi|)^{\otimes l} \right] \right\|_1 \leq 2l \mathop{\mathbb{E}}_{(|\varphi\rangle,|\psi\rangle)\sim\gamma_1} [\||\varphi\rangle - |\psi\rangle\|_2] \leq \frac{6l}{2^{\lambda n}}. \tag{13}$$

Finally, by the triangle inequality, Eqs. (12) and (13), we have

$$\left\| \mathop{\mathbb{E}}_{|\phi\rangle\sim\nu} \left[(|\phi\rangle\langle\phi|)^{\otimes l} \right] - \mathop{\mathbb{E}}_{|\psi\rangle\sim\mu} \left[(|\psi\rangle\langle\psi|)^{\otimes l} \right] \right\|_1 \leq \frac{40\pi(\lambda+1)nl}{\lambda^{\log\lambda} \cdot n^{\log n}} + \frac{6l}{2^{\lambda n}} = \mathrm{negl}(\lambda).$$

This establishes the *Statistical Pseudorandomness* property. □

5.3 Constructing PRSS

We construct a PRSS by replacing the random functions and permutations used in RSS with QPRFs and QPRPs.

Definition 10. *Let $n, T \in \mathbb{N}$, $\mathcal{H}$ be a real Hilbert space with dimension 2^n, $\tau : \mathcal{K}_1 \times \{0,1\}^n \to \{0,1\}^n$ be a* QPRP *with key space $\mathcal{K}_1$ and $F : \mathcal{K}_2 \times \{0,1\}^{n-1} \to \{0,1\}^d$ be a* QPRF *with key space $\mathcal{K}_2$. An ensemble of unitary operators $\mathsf{SG}^n := \left\{\mathsf{SG}^{n,\lambda}\right\}_\lambda$ with $\mathsf{SG}^{n,\lambda} := \left\{\mathsf{SG}_k^{n,\lambda} : \mathcal{H} \to \mathcal{H}\right\}_{k\in(\mathcal{K}_1\times\mathcal{K}_2)^T}$ is defined as*

$$\mathsf{SG}_k^{n,\lambda} = K_{\tau_{r_T},F_{s_T}} \cdots K_{\tau_{r_2},F_{s_2}} K_{\tau_{r_1},F_{s_1}}$$

for $k = (r_1, s_1, r_2, s_2, \ldots, r_T, s_T) \in (\mathcal{K}_1 \times \mathcal{K}_2)^T$, where $K_{\sigma,f} = U_{\sigma^{-1}} W_f U_\sigma$ is defined in (10).

Theorem 7. *Let $n \in \mathbb{N}$, $d = \log^2\lambda + \log^2 n$ and $T = 10(\lambda+1)n$. The ensemble of unitary operators SG^n defined in Definition 10 is a* PRSS.

Proof. Due to the efficiency of τ and F, the key length is bounded by $2T \cdot \mathrm{poly}(n, d) = \mathrm{poly}(n, \lambda)$. Thus the condition of polynomial-bounded key length is satisfied. To implement $\mathsf{SG}_k^{n,\lambda}$, we need to realize each of the $T = 10(\lambda+1)n$ unitary gates K that make up $\mathsf{SG}_k^{n,\lambda}$. Since each K can be realized in $\mathrm{poly}(n, \lambda)$ time (efficiency of τ and F), the overall construction time for $\mathsf{SG}_k^{n,\lambda}$ will be $\mathrm{poly}(n, \lambda)$. Thus the uniformity is also satisfied.

We now prove the pseudorandomness property. To this end, we consider three hybrids for an arbitrary $|\phi\rangle \in \mathcal{S}(\mathcal{H})$ and $l \in \mathrm{poly}(\lambda, n)$:

H1: $|\phi_k\rangle^{\otimes l}$ for $|\phi_k\rangle = \mathsf{SG}_k^{n,\lambda}|\phi\rangle$ where $k \leftarrow (\mathcal{K}_1 \times \mathcal{K}_2)^T$ is chosen uniformly at random.

H2: $\left|\varphi_{(\sigma_i)_{i=1}^T,(f_i)_{i=1}^T}\right\rangle^{\otimes l}$ for $\left|\varphi_{(\sigma_i)_{i=1}^T,(f_i)_{i=1}^T}\right\rangle = \mathsf{RSG}^{n,\lambda}_{(\sigma_i)_{i=1}^T,(f_i)_{i=1}^T}|\phi\rangle$ with independently and uniformly random permutations $(\sigma_i)_{i=1}^T \subseteq S_{2^n}$ and random functions $(f_i)_{i=1}^T \subseteq \{f : \{0,1\}^{n-1} \to \{0,1\}^d\}$. $\mathsf{RSG}^{n,\lambda}_{(\sigma_i)_{i=1}^T,(f_i)_{i=1}^T}$ is defined in Definition 9.

H3: $|\psi\rangle^{\otimes l}$ for $|\psi\rangle$ chosen according to the Haar measure μ on $\mathcal{S}^{2^n}_{\mathbb{R}}$.

We first prove that H1 and H2 are computationally indistinguishable. By the quantum-secure property of τ and F, we know the following two situations are computationally indistinguishable for any polynomial-time quantum oracle algorithm $\mathcal{A}$ (see Lemma 2):

- given oracle access to $\tau_{r_1}, \cdots, \tau_{r_T}$ and $F_{s_1}, \cdots, F_{s_T}$ where $(r_i)_{i=1}^T \subseteq \mathcal{K}_1$ and $(s_i)_{i=1}^T \subseteq \mathcal{K}_2$ are independently and uniformly random keys.
- given oracle access to independent and uniformly random permutations $(\sigma_i)_{i=1}^T \subseteq S_{2^n}$ and random functions $(f_i)_{i=1}^T \subseteq \{f : \{0,1\}^{n-1} \to \{0,1\}^d\}$.

Thus, we have for any polynomial-time quantum algorithm $\mathcal{A}$,

$$\left|\Pr\left[\mathcal{A}\left(|\phi_k\rangle^{\otimes l}\right)=1\right]-\Pr\left[\mathcal{A}\left(\left|\varphi_{(\sigma_i)_{i=1}^T,(f_i)_{i=1}^T}\right\rangle^{\otimes l}\right)=1\right]\right| = \mathrm{negl}(\lambda)\,.$$

For H2 and H3, they are statistically indistinguishable since RSG^n defined in Definition 9 is an RSS by Theorem 6. Finally, by the triangle inequality we establish H1 and H3 are computationally indistinguishable. This accomplishes the proof. □

5.4 The Dispersing Property

As mentioned before, our parallel Kac's walk also mixes rapidly in terms of total variation distance, which endows our scramblers with a unique dispersing property. We first introduce the concept of *dispersing random state scramblers* (DRSSs), which ensure the approximation of Haar randomness with respect to Wasserstein distance.

Definition 11 (Dispersing Random State Scrambler). *Let $\mathcal{H}_{\mathrm{in}}$ and $\mathcal{H}_{\mathrm{out}}$ be Hilbert spaces of dimensions 2^n and 2^m respectively with $n, m \in \mathbb{N}$ and $n \leq m$. Let $\mathcal{K} = \{0,1\}^\kappa$ be a key space, and λ be a security parameter. A* dispersing random state scrambler *(DRSS) is an ensemble of isometric operators $\mathcal{R}^{n,m} := \{\mathcal{R}^{n,m,\lambda}\}_\lambda$ with $\mathcal{R}^{n,m,\lambda} := \{\mathcal{R}^{n,m,\lambda}_k : \mathcal{H}_{\mathrm{in}} \to \mathcal{H}_{\mathrm{out}}\}_{k\in\mathcal{K}}$ satisfying:*

- ***Sphere Coverage.*** *There exist $\epsilon = \mathrm{negl}(\lambda)$ such that for any $|\phi\rangle \in \mathcal{S}(\mathcal{H}_{\mathrm{in}})$, the family of states $\{\mathcal{R}^{n,m}_k|\phi\rangle\}_{k\in\mathcal{K}}$ forms an ϵ-net of $\mathcal{S}(\mathcal{H}_{\mathrm{out}})$.*
- ***Wasserstein Approximation of Haar randomness.*** *There exist $\delta, \delta' = \mathrm{negl}(\lambda)$ such that for any $|\phi\rangle \in \mathcal{S}(\mathcal{H}_{\mathrm{in}})$,*
  - *Let ν be the distribution of $\mathcal{R}^{n,m}_k|\phi\rangle$ with uniformly random $k \leftarrow \mathcal{K}$, and μ be the Haar measure on $\mathcal{S}(\mathcal{H}_{\mathrm{out}})$. Then, there exists a distribution $\widetilde{\nu}$ such that*

$$\|\mu - \tilde{\nu}\|_{\mathrm{TV}} \leq \delta, \quad \textit{and} \quad W_\infty(\nu, \widetilde{\nu}) \leq \delta'.$$

– ***Uniformity***. *$\mathcal{R}^{n,m}$ can be uniformly computed in polynomial time. That is, there is a deterministic Turing machine that, on input $(1^n, 1^m, 1^\lambda, 1^\kappa)$, outputs a quantum circuit Q in $\mathrm{poly}(n, m, \lambda, \kappa)$ time such that for all $k \in \mathcal{K}$ and $|\phi\rangle \in \mathcal{S}(\mathcal{H}_{\mathrm{in}})$*

$$Q\,|k\rangle|\phi\rangle = |k\rangle|\phi_k\rangle\,,$$

where $|\phi_k\rangle := \mathcal{R}_k^{n,m,\lambda}|\phi\rangle$.

In particular, small Wasserstein distance implies small trace distance between the average states drawn from the two distributions. The proof is deferred to the full version.

Proposition 1. *A* DRSS *is an* RSS *with the same parameters.*

Moreover, we introduce a continuous version of random state scrambler, where continuous randomness is allowed.

Definition 12 (Continuously Random State Scrambler). *Let $\mathcal{H}_{\mathrm{in}}$ and $\mathcal{H}_{\mathrm{out}}$ be Hilbert spaces of dimensions 2^n and 2^m respectively with $n, m \in \mathbb{N}$ and $n \leq m$. Let $\mathcal{K}$ be a (continuous) key space, and λ be a security parameter. A* continuously random state scrambler *(*CRSS*) is an ensemble of isometric operators $\mathcal{R}^{n,m} := \{\mathcal{R}^{n,m,\lambda}\}_\lambda$ with $\mathcal{R}^{n,m,\lambda} := \{\mathcal{R}_k^{n,m,\lambda} : \mathcal{H}_{\mathrm{in}} \to \mathcal{H}_{\mathrm{out}}\}_{k\in\mathcal{K}}$ satisfying:*

– ***Total-Variation Approximation of Haar randomness.*** *Let $|\phi\rangle \in \mathcal{S}(\mathcal{H}_{\mathrm{in}})$ be an arbitrary pure state. Let ν be the distribution of $\mathcal{R}_k^{n,m,\lambda}|\phi\rangle$ with uniformly random $k \leftarrow \mathcal{K}$, and μ be the Haar measure on $\mathcal{S}(\mathcal{H}_{\mathrm{out}})$. Then there exists $\delta = \mathrm{negl}(\lambda)$ such that the total variation distance between ν and μ is at most δ, i.e., $\|\nu - \mu\|_{\mathrm{TV}} \leq \delta$.*

The following theorem proves that the RSS we construct over real space is also a DRSS.

Theorem 8. *Let $n \in \mathbb{N}$, $d = \log^2\lambda + \log^2 n$ and $T = 515(\lambda+1)n$. The ensemble of unitary operators RSG^n defined in Definition 9 is a* DRSS*.*

To prove Theorem 8, recall the ensemble of unitary operators $\widetilde{\mathsf{RSG}}^n := \left\{\widetilde{\mathsf{RSG}}^{n,\lambda}\right\}_\lambda$ we define in Sect. 5.2. We have the following proposition for $\widetilde{\mathsf{RSG}}^n$.

Proposition 2. *For $T = 515(\lambda+1)n$, the ensemble of unitary operator $\widetilde{\mathsf{RSG}}^n$ is a* CRSS*.*

Proof. Note that a uniformly random $\widetilde{\mathsf{RSG}}^{n,\lambda}_{(\sigma_i)_{i=1}^T,(\widetilde{f}_i)_{i=1}^T}$ corresponds to a T-step parallel Kac's walk on $\mathcal{S}_{\mathbb{R}}^{2^n}$. The proposition then follows from Theorem 5 and the definition of the CRSS. □

Let $|\eta\rangle \in \mathcal{S}(\mathcal{H})$ be an arbitrary real state. Set

$$\mathcal{N} = \left\{ \mathsf{RSG}^{n,\lambda}_{(\sigma_i)_{i=1}^T,(f_i)_{i=1}^T} |\eta\rangle \right\} \text{and} \widetilde{\mathcal{N}} = \left\{ \widetilde{\mathsf{RSG}}^{n,\lambda}_{(\sigma_i)_{i=1}^T,(\widetilde{f}_i)_{i=1}^T} |\eta\rangle \right\}. \tag{14}$$

We need the following two lemmas. Both proofs are deferred to the full version of this paper.

Lemma 5. $\widetilde{\mathcal{N}} = \mathcal{S}_{\mathbb{R}}^{2^n}$.

Lemma 6. *There exists an $\epsilon = \mathrm{negl}(\lambda)$ such that $\mathcal{N}$ is an ϵ-net for real vectors in $\mathcal{S}(\mathcal{H})$, where $\mathcal{N}$ is defined in (14).*

Proof of Theorem 8. It is easy to see that the uniformity condition is satisfied. Combining with Lemma 6, it suffices to prove that there exists a good distribution $\widetilde{\nu}$ satisfying the requirement in Definition 11. Fix $|\eta\rangle \in \mathcal{S}(\mathcal{H})$. Define three distributions:

- ν be the distribution of $\mathsf{RSG}^{n,\lambda}_{(\sigma_i)_{i=1}^T,(f_i)_{i=1}^T} |\eta\rangle$ with independent and uniformly random permutations $(\sigma_i)_{i=1}^T \subseteq S_{2^n}$ and random functions $(f_i)_{i=1}^T \subseteq \{f : \{0,1\}^{n-1} \to \{0,1\}^d\}$.
- $\widetilde{\nu}$ be the distribution of $\widetilde{\mathsf{RSG}}^{n,\lambda}_{(\sigma_i)_{i=1}^T,(\widetilde{f}_i)_{i=1}^T} |\eta\rangle$ with independent and uniformly random permutations $(\sigma_i)_{i=1}^T \subseteq S_{2^n}$, and random functions $(\widetilde{f}_i)_{i=1}^T \subseteq \{f : \{0,1\}^{n-1} \to [0,1)\}$.
- μ be the Haar measure on $\mathcal{S}_{\mathbb{R}}^{2^n}$.

Note that $\widetilde{\nu}$ is the output distribution of T-step parallel Kac's walk. Thus by Theorem 5, we have

$$\|\widetilde{\nu} - \mu\|_{\mathrm{TV}} \leq \frac{1}{2^{\lambda n - 1}} = \mathrm{negl}(\lambda). \tag{15}$$

We are left to show the Wasserstein ∞-distance between ν and $\widetilde{\nu}$ is negligible. To this end, we construct a coupling γ_0 of ν and $\widetilde{\nu}$ by using the same permutation σ_t and letting f_t be the function satisfying $f_t(y)$ is the d digits after the binary point in $\widetilde{f}_t(y)$ for all $y \in \{0,1\}^{n-1}$. When $(|v\rangle, |u\rangle) \sim \gamma_0$, we have

$$\begin{aligned} \||u\rangle - |v\rangle\|_2 &= \left\| \widetilde{\mathsf{RSG}}^{n,\lambda}_{(\sigma_i)_{i=1}^T,(\widetilde{f}_i)_{i=1}^T} |\eta\rangle - \mathsf{RSG}^{n,\lambda}_{(\sigma_i)_{i=1}^T,(f_i)_{i=1}^T} |\eta\rangle \right\|_2 \\ &\leq \left\| \widetilde{\mathsf{RSG}}^{n,\lambda}_{(\sigma_i)_{i=1}^T,(\widetilde{f}_i)_{i=1}^T} - \mathsf{RSG}^{n,\lambda}_{(\sigma_i)_{i=1}^T,(f_i)_{i=1}^T} \right\|_\infty \leq 2^{1-d}\pi T = \frac{1030\pi(\lambda+1)n}{\lambda^{\log\lambda} n^{\log n}}, \end{aligned} \tag{16}$$

where the last inequality is from Fact 1 and Lemma 4. Therefore,

$$\begin{aligned} W_\infty(\nu,\widetilde{\nu}) &= \lim_{p\to\infty} \left(\inf_{\gamma\in\Gamma(\nu,\widetilde{\nu})} \mathop{\mathbb{E}}_{(|v\rangle,|u\rangle)\sim\gamma} [\||v\rangle - |u\rangle\|_2^p] \right)^{1/p} \\ &\leq \lim_{p\to\infty} \left(\mathop{\mathbb{E}}_{(|v\rangle,|u\rangle)\sim\gamma_0} [\||v\rangle - |u\rangle\|_2^p] \right)^{1/p} \overset{\text{(Eq. (16))}}{\leq} \frac{1030\pi(\lambda+1)n}{\lambda^{\log\lambda} n^{\log n}} = \mathrm{negl}(\lambda). \end{aligned} \tag{17}$$

This completes the proof. □

6 Applications

Since pseudorandom state scramblers subsume pseudorandom state generators and its siblings in the literature, all applications enabled by PRSGs can also be obtained from PRSSs. This includes for instance symmetric-key encryption and commitment of classical messages as well as secure computation. In this section, we showcase a few novel applications beyond what PRSGs are capable of.

6.1 Compact Quantum Encryption

Because PRSSs map any initial state to a pseudorandom output state, we can readily employ them to encrypt quantum messages. Furthermore, it turns out that PRSS-based quantum encryption schemes offer improvements in terms of *compactness*, a point we discuss below.

We start by recalling the well-known Quantum One-Time Pad, which is the quantum analogue of one-time pad and achieves perfect secrecy. Given an n-qubit state $|\psi\rangle$, we sample a uniform $2n$-bit key $k = k_1 \| k_2$ with $k_1, k_2 \in \{0,1\}^n$ and encrypt $|\psi\rangle$ by

$$|\psi_k\rangle = \mathsf{QOTP}_k|\psi\rangle = X^{k_1} Z^{k_2} |\psi\rangle \ ,$$

where X and Z are Pauli operators applied on each qubit of $|\psi\rangle$.

We can reduce the key length by using pseudorandom keys. For instance, given a pseudorandom generator $\mathsf{PRG} : \{0,1\}^n \to \{0,1\}^{2n}$, we can expand a uniform n-bit key under PRG and use $\mathsf{PRG}(k)$ as the key to QOTP. Namely we encrypt by $|\psi_k\rangle = \mathsf{QOTP}_{\mathsf{PRG}(k)}|\psi\rangle$. We refer to this scheme as prg-QOTP.

These two schemes are secure if the same key is never used more than *once*. One can extend it to multi-time security with *hybrid* encryption, using in addition a post-quantum secure encryption for *classical* bits. For concreteness, we use a post-quantum $\mathsf{PRF}_k : \{0,1\}^n \to \{0,1\}^{2n}$. To encrypt $|\psi\rangle$, we sample a uniformly random string r, and use $\mathsf{PRF}_k(r)$ as the key to QOTP, i.e., we output cipherstate $(r, |\psi_{k,r}\rangle)$ where $|\psi_{k,r}\rangle = \mathsf{QOTP}_{\mathsf{PRF}_k(r)}|\psi\rangle$. We call this scheme prf-QOTP.

Now suppose we have a PRSS $(\{\mathcal{R}_k^{n,m}\})$ with key space $\mathcal{K} = \{0,1\}^\kappa$, and for simplicity we assume that $n = m$ and we ignore them in the notation. We can construct three encryption schemes, analogous to each of the schemes above (Table 2).

- PRSS-enc: on random key k and state $|\psi\rangle$, output

$$|\psi_k\rangle := \mathcal{R}_k|\psi\rangle \, .$$

- prg-PRSS-enc: given a $\mathsf{PRG} : \{0,1\}^n \to \mathcal{K}$, on random key k and state $|\psi\rangle$, output

$$|\psi_k\rangle := \mathcal{R}_{\mathsf{PRG}(k)}|\psi\rangle \, .$$

Table 2. Advantages of PRSS-based encryptions: maintaining single key instead of linear number of keys or reducing the cipher size growth factor by half.

ℓ copies of $\|\psi\rangle$	(prg-)QOTP	(prg-)PRSS-enc
$\ell = 1$	$\|\psi_k\rangle = \mathsf{QOTP}_{k \text{ or } \mathsf{PRG}(k)}\|\psi\rangle$	$\|\psi_k\rangle = \mathsf{PRSS}_{k \text{ or } \mathsf{PRG}(k)}\|\psi\rangle$
$\ell > 1$	$(\|\psi_{k_1}\rangle, \ldots, \|\psi_{k_\ell}\rangle)$	$(\|\psi_k\rangle, \ldots, \|\psi_k\rangle)$
Comparison	Need to exchange ℓ indep. keys $k_1, \ldots, k_\ell$	Single key for any polynomial ℓ
ℓ copies of $\|\psi\rangle$	prf-QOTP	prf-PRSS-enc
$\ell = 1$	$\left(r, \|\psi_{k,r}\rangle = \mathsf{QOTP}_{\mathsf{PRF}_k(r)}\|\psi\rangle\right)$	$(r, \|\psi_{k,r}\rangle) = \mathsf{PRSS}_{\mathsf{PRF}_k(r)}\|\psi\rangle$
$\ell > 1$	$\left(\ldots, \left(r_j, \|\psi_{k,r_j}\rangle\right), \ldots\right)$	$(r, \|\psi_{k,r}\rangle, \ldots, \|\psi_{k,r}\rangle)$
Comparison	Cipher size grows by ℓ factor	Cipher size grows by $\frac{1}{2}(\ell + 1)$

– prf-PRSS-enc: given a $\mathsf{PRF} : \{0,1\}^n \to \mathcal{K}$, the key is a random key k for the PRF. On state $|\psi\rangle$, output $(r, |\psi_{k,r}\rangle)$, where $r \leftarrow \{0,1\}^n$ and

$$|\psi_{k,r}\rangle = \mathcal{R}_{\mathsf{PRF}_k(r)}|\psi\rangle\,.$$

Advantages of PRSS*-Based Quantum Encryption.* One distinct benefit of PRSS-enc over QOTP is that we can encrypt *multiple copies* of a state $|\psi\rangle$ using PRSS-enc under the *same* key k. This follows from the multi-copy indistinguishability in our PRSS definition. In contrast, QOTP needs independent keys to encrypt each copy of $|\psi\rangle$. This considerably improves *compactness*, and it holds similarly in the other two types of schemes.

A related concept called *quantum private broadcasting* has been investigated by Broadbent, Gonzàlez-Guillén and Schuknecht [13]. They employ (symmetric) t-designs to encrypt t copies of an n-qubit quantum message. While the key length in their construction scales logarithmically with t, it grows exponentially with n. Our PRSS-based scheme maintains a key size of $\mathrm{poly}(n)$.

We stress that this applies only to encrypting multiple copies of the *same* input state. If we want to encrypt different states, then fresh keys in (prg-)PRSS-enc or randomness in prf-PRSS-enc should be used.

6.2 Succinct Quantum State Commitment

Next we show how PRSS enables quantum commitment. Bit commitment is a fundamental primitive in cryptography. A sender Alice commits to an input bit b to a receiver Bob in the *commit* phase, which can be revealed later in the *open* phase. This naturally extends to committing bit strings. Two properties are essential.

– *Hiding.* Bob is not able to learn the message b before the open phase.

– *Binding.* Alice cannot fool Bob to accept a different message $b' \neq b$ in the open phase.

We will focus on *non-interactive* commitment schemes where both commit and open phases consist of a single message from the sender to the receiver. If the protocol involves exchanging and processing quantum information, we call it a quantum bit commitment (QBC) scheme. QBC has been extensively studied, and it is shown that QBC can be constructed based on standard PRSGs [5,38].

In a similar vein, one can also consider committing to a *quantum* input state, and this is called quantum state commitment (QSC). QSC has proven useful such as in zero-knowledge proof systems for QMA [14,15].

Recently, Gunn, Ju, Ma and Zhandry give a systematic treatment on QSC [21]. They propose a new characterization of binding termed *swap-binding*. They show a striking hiding-binding *duality* theorem for (non-interactive) quantum commitment: binding holds if the *opening* register held by the sender hides the input state. This significantly simplifies proving binding. They then construct binding commitment schemes which in addition are *succinct*, where the register containing the commitment has a *smaller* size than the message state.[2]

Succinct QSC *from* PRSS. The succinct QSC schemes by [21] are based on post-quantum one-way functions or the potentially weaker primitive of pseudorandom unitary operators (PRU). We show below the viability of building succinct commitment on PRSS. Specifically, we observed that a *succinct* PRSS implies a succinct one-time quantum encryption, and it was shown in [21] that a succinct one-time quantum encryption gives a succinct QSC scheme. Hence (succinct one-time) PRSSs offer an alternate approach of realizing succinct one-time quantum encryptions based on potentially weaker assumptions than one-way functions, and could be weaker than the instantiation via PRUs in [21]. Meanwhile, one-time quantum encryption does not seem to follow immediately from PRS or other primitives implied by PRS.

Theorem 9. *Assuming a succinct* PRSS*, i.e.,* $|\mathcal{K}| < 2^n$*, there exists a succinct* QSC*.*

Proof. This follows from a generic claim in [21]. They show that any one-time secure quantum encryption scheme with *succinct* keys, where the key is shorter than the state to be encrypted, readily gives a succinct QSC. A PRSS is a secure quantum encryption as discussed above. Succinctness translates if PRSS's key length is shorter than the size of the input state. This is stated below. We choose not to fully spell out the syntax and definitions of the involved primitives for the sake of clarity, and refer the readers to [21]. □

Lemma 7. *Assuming a succinct* PRSS*, i.e.,* $|\mathcal{K}| < 2^n$*, there exists a succinct one-time quantum encryption scheme.*

[2] Note that hiding is not required in these succinct schemes.

How to instantiate a succinct PRSS? Our construction is not immediately succinct, because the key length $\Omega(\lambda \cdot n)$. We can remedy this by using a pseudorandom generator to expand a key shorter than n into pseudorandom keys for each iteration (QPRF and QPRP).

Acknowledgment. We thank the anonymous reviewers' feedback. CL and FS were supported in part by the US National Science Foundation grants CCF-2042414, CCF-2054758 (CAREER) and CCF-2224131. MQ, PY and MZ were supported in part by National Natural Science Foundation of China (Grant No. 62332009, 61972191), and Innovation Program for Quantum Science and Technology (Grant No. 2021ZD0302901). FS thanks Zhengfeng Ji and Yi-Kai Liu for discussions on the topic of quantum pseudorandomness.

References

1. Aaronson, S., et al.: Quantum pseudoentanglement. In: 15th Innovations in Theoretical Computer Science Conference (ITCS 2024). Leibniz International Proceedings in Informatics (LIPIcs), vol. 287, pp. 2:1–2:21. Schloss Dagstuhl – Leibniz-Zentrum für Informatik (2024). https://doi.org/10.4230/LIPIcs.ITCS.2024.2
2. Alagic, G., Majenz, C., Russell, A.: Efficient simulation of random states and random unitaries. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020. LNCS, vol. 12107, pp. 759–787. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-45727-3_26
3. Ananth, P., Gulati, A., Kaleoglu, F., Lin, Y.T.: Pseudorandom isometries. In: Advances in Cryptology – EUROCRYPT 2024, pp. 226–254. Springer, Cham (2024)
4. Ananth, P., Gulati, A., Qian, L., Yuen, H.: Pseudorandom (function-like) quantum state generators: new definitions and applications. In: Kiltz, E., Vaikuntanathan, V. (eds.) TCC 2022, Part I, pp. 237–265. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-22318-1_9
5. Ananth, P., Qian, L., Yuen, H.: Cryptography from pseudorandom quantum states. In: Dodis, Y., Shrimpton, T. (eds.) CRYPTO 2022, Part I, pp. 208–236. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-15802-5_8
6. Behera, A., Sattath, O.: Almost public quantum coins. arXiv preprint arXiv:2002.12438 (2020)
7. Bouland, A., Fefferman, B., Vazirani, U.: Computational pseudorandomness, the wormhole growth paradox, and constraints on the AdS/CFT duality. In: 11th Innovations in Theoretical Computer Science Conference (ITCS 2020). Leibniz International Proceedings in Informatics (LIPIcs), vol. 151, pp. 63:1–63:2. Schloss Dagstuhl – Leibniz-Zentrum für Informatik (2020). https://doi.org/10.4230/LIPIcs.ITCS.2020.63
8. Brakerski, Z., Magrafta, N.: Real-valued somewhat-pseudorandom unitaries (2024)
9. Brakerski, Z., Shmueli, O.: (Pseudo) Random quantum states with binary phase. In: Hofheinz, D., Rosen, A. (eds.) TCC 2019. LNCS, vol. 11891, pp. 229–250. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-36030-6_10
10. Brakerski, Z., Shmueli, O.: Scalable pseudorandom quantum states. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020. LNCS, vol. 12171, pp. 417–440. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-56880-1_15

11. Brandão, F.G., Chemissany, W., Hunter-Jones, N., Kueng, R., Preskill, J.: Models of quantum complexity growth. PRX Quantum **2**, 030316 (2021). https://doi.org/10.1103/PRXQuantum.2.030316
12. Brandão, F.G., Harrow, A.W., Horodecki, M.: Local random quantum circuits are approximate polynomial-designs. Commun. Math. Phys. **346**, 397–434 (2016). https://doi.org/10.1007/s00220-016-2706-8
13. Broadbent, A., González-Guillén, C.E., Schuknecht, C.: Quantum private broadcasting. Phys. Rev. A **105**, 022606 (2022). https://doi.org/10.1103/PhysRevA.105.022606
14. Broadbent, A., Grilo, A.B.: QMA-hardness of consistency of local density matrices with applications to quantum zero-knowledge. SIAM J. Comput. **51**(4), 1400–1450 (2022). https://doi.org/10.1137/21M140729X
15. Broadbent, A., Ji, Z., Song, F., Watrous, J.: Zero-knowledge proof systems for QMA. SIAM J. Comput. **49**(2), 245–283 (2020). https://doi.org/10.1137/18M1193530
16. Chatterjee, S., Diaconis, P., Sly, A., Zhang, L.: A phase transition for repeated averages. Ann. Probab. **50**(1), 1–17 (2022). https://doi.org/10.1214/21-AOP1526
17. Chen, C.F., Bouland, A., Brandão, F.G.S.L., Docter, J., Hayden, P., Xu, M.: Efficient unitary designs and pseudorandom unitaries from permutations (2024)
18. Coladangelo, A.: Quantum trapdoor functions from classical one-way functions. arXiv preprint arXiv:2302.12821 (2023)
19. Dankert, C., Cleve, R., Emerson, J., Livine, E.: Exact and approximate unitary 2-designs and their application to fidelity estimation. Phys. Rev. A **80**(1), 012304 (2009). https://doi.org/10.1103/PhysRevA.80.012304
20. Diaconis, P., Saloff-Coste, L.: Bounds for Kac's master equation. Commun. Math. Phys. **209**, 729–755 (2000). https://doi.org/10.1007/s002200050036
21. Gunn, S., Ju, N., Ma, F., Zhandry, M.: Commitments to quantum states. In: Proceedings of the 55th Annual ACM Symposium on Theory of Computing, pp. 1579–1588. Association for Computing Machinery (2023). https://doi.org/10.1145/3564246.3585198
22. Haferkamp, J.: Random quantum circuits are approximate unitary t-designs in depth $O\left(nt^{5+o(1)}\right)$. Quantum **6**, 795 (2022). https://doi.org/10.22331/q-2022-09-08-795
23. Harrow, A.W., Low, R.A.: Efficient quantum tensor product expanders and k-designs. In: Dinur, I., Jansen, K., Naor, J., Rolim, J. (eds.) APPROX/RANDOM -2009. LNCS, vol. 5687, pp. 548–561. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-03685-9_41
24. Harrow, A.W., Mehraban, S.: Approximate unitary t-designs by short random quantum circuits using nearest-neighbor and long-range gates. Commun. Math. Phys. **401**(2), 1531–1626 (2023). https://doi.org/10.1007/s00220-023-04675-z
25. Hastings, W.K.: Monte Carlo sampling methods using Markov chains and their applications. Biometrika **57**(1), 97–109 (1970). https://doi.org/10.1093/biomet/57.1.97
26. Haug, T., Bharti, K., Koh, D.E.: Pseudorandom unitaries are neither real nor sparse nor noise-robust. arXiv preprint arXiv:2306.11677 (2023)
27. Hough, B., Jiang, Y.: Cut-off phenomenon in the uniform plane Kac walk. Annals Prob. **45**(4), 2248–2308 (2017). http://www.jstor.org/stable/26362254
28. Jain, V., Pillai, N.S., Sah, A., Sawhney, M., Smith, A.: Fast and memory-optimal dimension reduction using Kac's walk. Ann. Appl. Probab. **32**(5), 4038–4064 (2022). https://doi.org/10.1214/22-AAP1784

29. Janvresse, É.: Bounds on semigroups of random rotations on SO(n). Theory Prob. Appl. **47**(3), 526–532 (2003). https://doi.org/10.1137/S0040585X97979950
30. Ji, Z., Liu, Y.-K., Song, F.: Pseudorandom Quantum States. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018. LNCS, vol. 10993, pp. 126–152. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-96878-0_5
31. Jiang, Y.: Total variation bound for Kac's random walk. In: The Annals of Applied Probability, pp. 1712–1727 (2012). https://www.jstor.org/stable/41713374
32. Kac, M.: Foundations of kinetic theory. In: Third Berkeley symposium on mathematical statistics and probability. vol. 3, pp. 171–197 (1956)
33. Kim, I., Tang, E., Preskill, J.: The ghost in the radiation: robust encodings of the black hole interior. J. High Energy Phys. **2020**(6), 1–65 (2020). https://doi.org/10.1007/JHEP06(2020)031
34. Kretschmer, W.: Quantum pseudorandomness and classical complexity. In: 16th Conference on the Theory of Quantum Computation, Communication and Cryptography – TQC 2021, pp. 2:1–2:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik (2021). https://doi.org/10.4230/LIPIcs.TQC.2021.2
35. Kretschmer, W., Qian, L., Sinha, M., Tal, A.: Quantum cryptography in algorithmica. In: 55th Annual ACM Symposium on Theory of Computing (STOC 2023). pp. 1589–1602 (2023). https://doi.org/10.1145/3564246.3585225
36. Lu, C., Qin, M., Song, F., Yao, P., Zhao, M.: Quantum pseudorandom scramblers. Cryptology ePrint Archive, Paper 2024/1470 (2024). https://eprint.iacr.org/2024/1470
37. Metger, T., Poremba, A., Sinha, M., Yuen, H.: Simple constructions of linear-depth t-designs and pseudorandom unitaries (2024)
38. Morimae, T., Yamakawa, T.: Quantum commitments and signatures without one-way functions. In: Advances in Cryptology – CRYPTO 2022, pp. 269–295. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-15802-5_10
39. Movassagh, R., Szegedy, M., Wang, G.: Repeated averages on graphs. arXiv:2205.04535 (2022)
40. O'Donnell, R., Servedio, R.A., Paredes, P.: Explicit orthogonal and unitary designs. In: 2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS), pp. 1240–1260. IEEE Computer Society (2023). https://doi.org/10.1109/FOCS57990.2023.00073
41. Oliveira, R.I.: On the convergence to equilibrium of Kac's random walk on matrices. Annals Appl. Probab. **19**(3), 1200–1231 (2009). https://www.jstor.org/stable/30243617
42. Pillai, N.S., Smith, A.: Kac's walk on n-sphere mixes in $n \log n$ steps. Ann. Appl. Probab. **27**(1), 631–650 (2017). https://doi.org/10.1214/16-AAP1214
43. Pillai, N.S., Smith, A.: On the mixing time of Kac's walk and other high-dimensional Gibbs samplers with constraints. Annals Probab. **46**(4), 2345–2399 (2018). https://www.jstor.org/stable/26506605
44. Renes, J.M., Blume-Kohout, R., Scott, A.J., Caves, C.M.: Symmetric informationally complete quantum measurements. J. Math. Phys. **45**(6), 2171–2180 (2004). https://doi.org/10.1063/1.1737053
45. Yang, L., Engelhardt, N.: The complexity of learning (pseudo)random dynamics of black holes and other chaotic systems. arXiv preprint arXiv:2302.11013 (2023)
46. Zhandry, M.: A note on quantum-secure PRPs. Cryptology ePrint Archive, Paper 2016/1076 (2016). https://eprint.iacr.org/2016/1076
47. Zhandry, M.: How to construct quantum random functions. J. ACM **68**(5), 1–43 (2021). https://doi.org/10.1145/3450745

Real-Valued Somewhat-Pseudorandom Unitaries

Zvika Brakerski and Nir Magrafta(✉)

Weizmann Institute of Science, Rehovot, Israel
{zvika.brakerski,nir.magrafta}@weizmann.ac.il

Abstract. We explore a very simple distribution of unitaries: random (binary) phase—Hadamard—random (binary) phase—random computational-basis permutation. We show that this distribution is statistically indistinguishable from random Haar unitaries for any polynomial set of orthogonal input states (in any basis) with polynomial multiplicity. This shows that even though real-valued unitaries cannot be completely pseudorandom (Haug, Bharti, Koh, arXiv:2306.11677), we can still obtain some pseudorandom properties without giving up on the simplicity of a real-valued unitary.

Our analysis shows that an even simpler construction: applying a random (binary) phase followed by a random computational-basis permutation, would suffice, assuming that the input is orthogonal and *flat* (that is, has high min-entropy when measured in the computational basis).

Using quantum-secure one-way functions (which imply quantum-secure pseudorandom functions and permutations), we obtain an efficient cryptographic instantiation of the above.

1 Introduction

Pseudorandomness is one of the most fundamental notions in cryptography. Prominent examples include pseudorandom generators (PRG) [11], pseudorandom functions (PRF) [10], and pseudorandom permutations (PRP) [16], which play a crucial role in various constructions in cryptography and beyond. Let us consider the concept of PRP, which is quite analogous to the object at the focus of this work. If we consider the class of all permutations $\{0,1\}^n \to \{0,1\}^n$, a random function from this class requires an exponential number of random bits to specify, and requires an exponential-size circuit to evaluate (and invert). A PRP is a distribution that can be sampled using a *polynomial* number of bits, known as the *seed*.[1] Furthermore, given the seed s, it is possible to evaluate and invert the associated permutation π_s in polynomial time. The crucial point is that any polynomial time process cannot distinguish between an interaction with

[1] In a formal definition, one has to address the subtlety of whether "efficiency" is defined with respect to the input length n, or with respect to some "security parameter". This distinction does not matter for our current discussion, and we will point out when it does down the line.

For the most up-to-date version of this work, please refer to https://arxiv.org/2403.16704.

E. Boyle and M. Mahmoody (Eds.): TCC 2024, LNCS 15365, pp. 36–59, 2025.
https://doi.org/10.1007/978-3-031-78017-2_2

a permutation π_s for a random seed s, and an interaction with a completely random permutation. It has been established [10,11,16] that PRG, PRF and PRP can all be constructed given the existence of one-way functions: the most basic (classical) cryptographic primitive. Furthermore, this connection is true even if we consider a quantum (polynomial-time) adversary [18,19].

In the context of quantum computing and quantum cryptography, Ji Liu and Song [14] (henceforth JLS) proposed to study pseudorandomness of quantum objects. In particular, the defined the notion of *pseudorandom quantum states* (PRS) which are n-qubit states which are (indistinguishable from) Haar random n-qubit states, even with arbitrary polynomial-time interaction with a polynomial number of copies of the pseudorandom state. The notion of PRS has been the subject of extensive study since [3–7,9,13].

Another object proposed by JLS is that of pseudorandom unitaries (PRU). Similarly to PRP, these should allow to evaluate and invert a unitary given a seed of polynomially many random bits, while being computationally indistinguishable from a random Haar unitary given arbitrary polynomial time interaction. Contrary to PRS, JLS presented constructions of a PRU, but were unable to prove their security. To date, it is still unknown how to construct PRU with a security proof under any known cryptographic assumption.

Recently, some partial progress has been made towards a construction of PRU. Namely, several works introduced families of unitaries with polynomial seeds and efficient evaluation, but falling short on pdeudorandomness. Lu, Qin, Song, Yao, and Zhao [15] introduced the notion of Pseudorandom State Scramblers (PRSS), which are unitaries that are only proven to act pseudorandomly on an arbitrary *single input state* with arbitrary polynomial multiplicity. Namely, any number of copies of the output on one particular input are pseudorandom. Ananth, Gulati, Kaleoglu, and Lin [2] introduced the notion of Pseudorandom Isometries (PRI), which are not unitary since their output is longer than their input (and the security of the constructions hinges on this property). They proved security of the PRI property for the case of a single input, a polynomial number of Haar-random inputs, or for an inputs which are a subset of computational basis elements (all with arbitrary polynomial multiplicity). The works of [2,15] mention a number of applications for the primitives that they defined, including multi-copy security for quantum public-key encryption.

Interestingly, the constructions in [2,15] are *real-valued*. Namely, the unitary family consists of unitaries with only real values.[2] In contrast, Haug, Bharti, and Koh [12] showed that *full PRU security* cannot be achieved in this way. This is done by observing that if U is real valued, then $U \otimes U$ acts as identity on the maximally entangled state, whereas this is very far from being the case if U is a random unitary. Therefore, if we consider adversaries that are allowed to make entangled queries to the unitary, then it is impossible to construct real-valued pseudorandom unitaries. Indeed, very recently, and concurrently and independently of our work, Metger, Poremba, Sinha, and Yuen [17] and Chen, Bouland,

[2] In [15] there is an additional construction which uses complex unitaries, but for the PRSS property, a real valued construction suffices.

Brandão, Docter, Hayden, and Xu [8] showed that it is possible to construct *non-adaptive PRU* that are pseudorandom with respect to any set of inputs that cannot change adaptively throughout the querying process, even when those inputs are entangled with each other and/or the environment. Indeed, their construction is inherently complex-valued. The question, therefore, remains:

What pseudorandom properties can be shown for real-valued efficiently computable unitaries?

In this work, we show that it is possible to achieve stronger security notions than [2,15] using an extremely simple construction.

1.1 Our Results

We consider an extremely simple family of unitaries: $U_P U_G H^{\otimes n} U_F$, where for functions $F, G : \{0,1\}^n \to \{\pm 1\}$, the operators U_F, U_G are the unitaries $|x\rangle \to F(x)|x\rangle, |x\rangle \to G(x)|x\rangle$, and for a permutation $P : \{0,1\}^n \to \{0,1\}^n$, U_P is the unitary $|x\rangle \to |P(x)\rangle$. The functions F, G in our construction are quantum-secure pseudorandom functions, and the permutation P is a quantum-secure pseudorandom permutation. Using the security of F, G and P, we can replace them with truly random counterparts f, g and π. We then analyze the output of the construction information theoretically with f, g and π. We note that this construction is in the spirit of, but simpler than, the PRU candidates considered by [14].

We show that our family of unitaries acts as a PRU so long as the inputs are (a mixture of) an *orthogonal* set of quantum states, with arbitrary polynomial multiplicity each. This in particular shows that this construction is also a PRSS. Our construction also generalizes the properties proven by [2] for PRI, without increasing the output size and using a construction of comparable complexity.

Notably, our construction can be separated into two parts, each of which is interesting in its own right. First, we show that $H^{\otimes n} U_f$ is a "state-flattener", in the sense that for any polynomial size set of input states, it holds that with overwhelming probability over a truly random function f, the output states are all "almost perfectly flat" in the computational basis. Namely, the square-magnitude of each computational basis element is bounded by $\epsilon = O(\frac{n}{2^n})$ (note that $1/2^n$ is the maximum possible flatness, and Haar random states are also expected to have $\sim \frac{n}{2^n}$ flatness). This property follows immediately from known concentration bounds, but we believe that it was not explicitly pointed out in this context. So, if we only want to approximate the flattening property of PRU, it can be done almost trivially.

We then show that the second part of our construction, $U_\pi U_g$ for random π, g, acts as PRU for flat orthogonal inputs. Again, this is an extremely simple construction that can be applied even as-is for a non-trivial set of input states (e.g. some polynomial subset of a random basis for the given Hilbert space). The technical crux of our paper is in the analysis of this component.

1.2 Technical Overview

We provide an overview of the proof for our main information theoretic lemma. That is, taking f, g, π to be random functions and a permutation, then our construction is statistically indistinguishable from a Haar random unitary for orthogonal inputs.

We use an (approximate) characterization of the output of querying a Haar random unitary on orthogonal input states. For t copies of s different orthogonal inputs, the output "target" state can be approximated by the density matrix

$$\rho_{\text{target}} = \sum_{z,\sigma} |z\rangle\langle\sigma(z)| \ , \tag{1}$$

ignoring global normalization. The summation is over all $z \in (\{0,1\}^n)^{st}$ whose st entries are unique elements in $\{0,1\}^n$, and over a set of permutations σ over the set $[st]$ (or, equivalently, over $[t]^s$). Namely, the permutation σ takes a vector $z \in (\{0,1\}^n)^{st}$ and permutes its entries (the vector $\sigma(z)$ has the same set of entries as z, only in a different order). Specifically, the summation is only over "block-preserving" permutations, which are permutations that only swap elements inside each t-tuples of elements. That is, a permutation is block-preserving if it can be represented as a sequence of s permutations over $[t]$.

We then prove that the output of our construction, with random functions and permutation, is close to the state in Eq. (1), thereby showing that on our set of input states, our construction is statistically indistinguishable from a Haar random unitary.

In this overview we first explain how to prove the flatness property for the first part of our construction, and then consider the second part of our construction. The former is described in Sect. 1.2. For the latter, we first explain in Sect. 1.2 how to "clean up" the state by removing cross-terms and certain "asymmetric" terms. This part is similar in spirit to what is done in previous works (although we present a more general analysis that is based only on flatness and not on specific properties of the input state). Then, we are left with the most technically involved part which is to analyze bound the trace norm of the difference between our state (call it ρ_{sym}) and the state ρ_{target} above. This is explained in Sect. 1.2.

Flattening. Recall that we consider the unitary distribution $H^{\otimes n}U_f$, where f is a random function (i.e. a random binary phase followed by Hadamard on all qubits). We say that a vector is ϵ-flat if the square-absolute-value of each of its (standard basis) coefficients is bounded by ϵ.

Given an input state of the form $\beta = \sum_x \beta_x |x\rangle$, we consider $\gamma_y = \langle y|H^{\otimes n}U_f|\beta\rangle$, which is the amplitude of the standard basis element $|y\rangle$ in the vector $H^{\otimes n}U_f|\beta\rangle$. This value can be expressed as an exponential sum $\gamma_y = \frac{1}{2^{n/2}} \sum_x f(x)(-1)^{x\cdot y}\beta_x$. We interpret each summand as a random variable with zero mean, since f is a random function to $\{\pm 1\}$. This means that we have a sum of exponentially many independent zero-mean random variables, and furthermore the ℓ_2 norm of the vector of summands is bounded since $\sum_x |\beta_x|^2 = 1$.

This means that the sum is very strongly concentrated around 0, and indeed using Hoeffding, the square-absolute-value will be at most $\frac{cn}{2^n}$ with all but an exponentially small probability. By applying the union bound, we get that for any a-priori polynomial-size set of input vectors, and for any coefficient γ_y of any of these vectors, it holds with all but exponentially small probability that they are all bounded by $\frac{cn}{2^n}$ in square-absolute-value. We note that since we consider complex vectors, the actual analysis separates β into its real and imaginary part, and analyze each separately.

From this point and on, we analyze the remainder of our construction under the assumption that the input quantum states are $\frac{cn}{2^n}$-flat.

Cross-Term Removal and Symmetrization. We consider the application of $(U_\pi U_g)^{\otimes st}$ on an input state consisting of s blocks, each of which contains t copies of the same (flat) state, where the vectors in the different blocks are orthogonal. Our goal is to show that, up to normalization, the output state can be expressed as

$$\rho_{\text{sym}} = \sum_{z,\sigma} \nu_\sigma |z\rangle\langle\sigma(z)| \ , \tag{2}$$

where z is summed as in Eq. (1), σ ranges over *all* permutations of $[st]$ (not only block-preserving ones), and ν_σ is a term that will be explained below.

As mentioned above, the techniques here are fairly standard in the analysis of pseudorandom states and other objects [3]. However, our analysis relies only on the general notion of flatness and not on the specific expression for the coefficients of the state.

We start by using the flatness of the states. Let $\Pi^\star$ be the projector to the vectors of length st of strings in $\{0,1\}^n$ with unique entries, that is, no entry reoccurs. Then our input state is close to its $\Pi^\star$ projection, since the collision probability in the standard basis of two entries is small due to flatness. Therefore, we may consider an input state whose density matrix supported only over entries $|z\rangle\langle z'|$, where both z, z' are unique st-tuples of elements from $\{0,1\}^n$.

We first apply $U_g^{\otimes st}$, which has the effect of zeroing out the coefficients of $|z\rangle\langle z'|$ not of the form $|z\rangle\langle\sigma(z)|$. This is the result of $\mathbb{E}_g\left[\prod_i g(z_i)\prod_i g(z'_i)\right]$ being zero for all z, z' which do not have the same entry-histogram since we average over random g's.

Finally, applying $U_\pi^{\otimes st}$ means that the coefficient of $|z\rangle\langle\sigma(z)|$ becomes independent of z, and depends only on σ. This follows from taking the expectation over π, which averages the coefficients $|z\rangle\langle\sigma(z)|$ for all unique entries z (since π is a random permutation). We denote the coefficient corresponding to σ by ν_σ.

Bounding The Difference. The difference between Eq. (1) and Eq. (2) is two-fold. First, the target state only sums over block-preserving permutations, whereas the symmetrized state sums over all permutations. Second, the target state gives the same weight to all terms $|z\rangle\langle\sigma(z)|$ that it ranges over, whereas the symmetrized state may give different weights to different permutations.

Our crucial observation here, is to notice that if it is possible to go from a permutation σ to a permutation σ' by performing block-preserving operations, then σ and σ' have the same coefficient ν_σ. This is the case since the input state corresponds to s blocks where each block contains t identical states. Therefore, the state of the system should be invariant under block-preserving permutations, which is manifested in the corresponding terms ν_σ being equal. We may therefore define congruence classes of permutations that differ only by block-preserving operations, and associate the coefficients with the congruence class rather than a specific permutation.

Consider for every congruence class of permutations p the operator $A_\mathsf{p} = \sum_{\sigma\in\mathsf{p}}\sum_z |z\rangle\langle\sigma(z)|$. Then we can rewrite $\rho_{\text{sym}} = \sum_\mathsf{p} \nu_\mathsf{p} A_\mathsf{p}$. We note that the set of all block-reserving permutations consists of a single congruence class. Therefore, all block-preserving permutations receive the same weight, as required. The remaining goal is to show that the classes that correspond to permutations that are not block-preserving are (jointly) negligible in ℓ_1 weight.

We let k be the number of "crossings" of a congruence class. That is, the number of inputs whose output belongs to a different block. The crossing number, in some sense, represents the amount of deviation from being block-preserving. Denote by p_k the set of congruence classes with k crossings. Note that there is a single congruence class with zero crossings, and it corresponds to the aforementioned set of block-preserving permutations. Our goal therefore is to bound the trace norm of $\sum_{k>1}\sum_{\mathsf{p}\in\mathsf{p}_k}\nu_\mathsf{p}A_\mathsf{p}$.[3]

The argument here contains three parts:

1. We show that for combinatorial reasons, $\|A_\mathsf{p}\|$ grows with $\text{poly}(nst)^k$ (up to a global normalization factor). Essentially, we show that this norm is related to the number of permutations in p (which due to symmetry is the same in all p with the same k).
2. We show, again by a combinatorial argument, that the number of congruence classes with the same k also grows as $\text{poly}(nst)^k$.
3. Perhaps the most technically involved part is to show that ν_p decays, up to a global normalization factor, with δ^k for a (negligible) factor $\delta = \frac{\text{poly}(nst)}{2^n}$. This is achieved by noticing that ν_p contains an "inner product" term for every crossing edge. This term would ideally correspond to an inner product between two input vectors, and since these are orthogonal we would expect this value to be 0. However, the inner product is "disturbed" because permutations do not allow recurrence, which in turn creates dependence between the would-be inner products. We therefore need to come up with a fairly involved technical argument to show that whereas the value ν_p is not exactly 0, each of the would-be inner products contributes a δ factor, resulting in an exponential decay.

Other factors cancel out perfectly, since they represent the same combinatorial reality, and indeed when putting-together all of the above, we get sum of

[3] We notice that $k = 1$ is not possible for reasons of symmetry and therefore it does not appear in the sum, but we could have achieved our result even without this minor optimization.

the form $\sum_{k>1}\left(\frac{\text{poly}(nst)}{2^n}\right)^k$, which converges to a negligible value as required, bearing in mind that $s, t = \text{poly}(n)$.

1.3 Other Previous Works

Alagic, Majenz, and Russell [1] considered a *stateful* variant of PRU, where the adversary is interacting with a *stateful simulator* whose internal state may change and grow as the experiment proceeds. Even under this relaxed notion, they were only able to construct PRU using a simulator in (quantum) PSPACE. However, in this variant one can hope to achieve *statistical* (or even perfect) security rather than just computational.

1.4 Future Directions

Contrary to the constructions in [15], ours is not *scalable* (a notion introduced in [7]). That is, we consider adversaries whose running time is polynomial in the input size of the unitary n. In contrast, in a scalable construction, one specifies separately the parameters for the adversary's complexity and for the input size. Scalable constructions are usually more involved, and in particularly require more computational depth, than non-scalable ones. It remains an open question to find a scalable version of our construction, or to prove lower bounds in this vein.

Our work shows a fairly strong notion of pseudorandomness for PRU that can be achieved using a straightforward real-valued construction. One may further wonder whether it is possible to get even closer to full-fledged PRU using constructions like ours (or even our construction as-is). For example, the negative result of [12] does not seem to exclude real-valued PRU that are applicable to *tensor-product* inputs. Namely, inputs that are not necessarily orthogonal, but are not entangled with each other (recall that entangled queries stand at the core of the [12] separation). We believe that answering some of these questions may be within reach, but were hurried to report our current progress due to the recent announcement of [17].

1.5 Paper Organization

Section 2 introduces notations, defines quantum-secure pseudorandom functions and permutations, and references the notion of almost invariancy under Haar random unitaries. Section 3 provides the security definition for non-adaptive orthogonal-inputs pseudorandom unitaries, presents the main theorem, describes the construction and reduce it to the information theoretic version. Section 4 contains the technical contributions. It starts with the main information theoretic lemma, continues with the analysis of the two steps of the construction separately, and concludes with proving the main lemma and theorem.

2 Preliminaries

2.1 Notation

We denote $N = 2^n$ and $N^{\underline{st}} = \binom{N}{st}(st)! = N(N-1)\cdots(N-st+1)$. We denote the trace norm of A by $\|A\|_1 = \mathrm{Tr}(\sqrt{AA^\dagger})$, which is the sum of the singular values.

Vectors, Functions, and Permutations. Let $s, t \in \mathbb{N}$. Throughout our analysis we will consider vectors of bit-strings. Formally, an object of the form $\boldsymbol{y} \in (\{0,1\}^n)^{st}$, which indicates a t-length vector of n-bit strings. We will denote $\boldsymbol{y} = (y_1, \ldots, y_{st})$, where each coordinate in the vector is a bit-string $y_i \in \{0,1\}^n$. Let T be some set. We denote by $\mathcal{U}^\star_{n,st}$ the set of all length st vectors of *unique* elements from $\{0,1\}^n$, that is, no entry reoccurs.

We will consider a number of types of operations on such vectors. For any function $f : \{0,1\}^n \to D$, where D is some domain, we let $f(\boldsymbol{y}) \in D^t$ denote the pointwise application of f on each coordinate in $\boldsymbol{y}$ individually. For functions with complex range $\alpha : \{0,1\}^n \to \mathbb{C}$, we also define multiplicative notation, where we use the notation $\alpha_x = \alpha(x)$, and $\alpha_{\boldsymbol{x}} = \prod_i \alpha_{x_i}$. Note that the coefficients of quantum states constitute such functions.

For the special case of a function on the coordinates which is a permutation, we will denote it by $\pi : \{0,1\}^n \to \{0,1\}^n$. We call such a permutation an *inner permutation* since it permutes each element of $\boldsymbol{y}$ individually. We also consider an *outer permutation* (or index permutation) $\sigma \in S_{st}$ which permutes $[st] = \{1, \ldots, st\}$. We will abuse the notation and think of $\sigma \in S_{st}$ also as $\sigma \in S_{[s]\times[t]}$ which takes a tuple input $(j,i), j \in [s], i \in [t]$ and outputs the tuple (j', i') s.t. $j' = \lfloor \sigma(sj+i)/s \rfloor, i' = \sigma(sj+i) \mod t$. An outer permutation permutes the indices of the elements of $\boldsymbol{y}$, i.e. $\boldsymbol{z} = \sigma(\boldsymbol{y}) \in (\{0,1\}^n)^{st}$ is such that $z_{j,i} = y_{\sigma(j,i)}$. We will also consider the subgroup $S_t^s = S_t \times \cdots \times S_t$ (s times) of outer permutations.

2.2 Concentration Bounds

Theorem 2.1 (Hoeffding's Inequality). *Let Z_x, $x \in \{0,1\}^n$, be independent random variables with zero expectation such that $a_x \le Z_x \le b_x$ with probability 1. Then for all $\epsilon > 0$,*

$$\Pr\left[\sum_{x\in\{0,1\}^n} Z_x \ge \epsilon\right] \le \exp\left(\frac{-2\epsilon^2}{\sum_i (b_i - a_i)^2}\right). \tag{3}$$

2.3 Pseudorandomness

Zhandry proved that given quantum-secure one-way functions, we can construct quantum-secure pseudorandom functions [18] and pseudorandom permutations [19], which we use in our construction.

Definition 2.2 (Quantum-Secure Pseudorandom Function). *Let $\mathcal{K}$ be a key space. A keyed family of functions $\{F_k : \{0,1\}^n \to \{\pm 1\}\}_{k\in\mathcal{K}}$ is a quantum-secure pseudorandom function (QPRF) if for any (non-uniform) quantum polynomial-time (QPT) oracle algorithm $\mathcal{A}$, F_k with a random $k \leftarrow \mathcal{K}$ is indistinguishable from a truly random function f in the sense that*

$$\left|\Pr_k[\mathcal{A}^{F_k}(1^n) = 1] - \Pr_f[\mathcal{A}^f(1^n) = 1]\right| = \mathrm{negl}(n).$$

In addition, F_k is polynomial-time computable on a classical computer.

Definition 2.3 (Quantum-Secure Pseudorandom Permutation). *Let $\mathcal{K}$ be a key space. A keyed family of permutations $\{P_k : \{0,1\}^n \to \{0,1\}^n\}_{k\in\mathcal{K}}$ is a quantum-secure pseudorandom permutation (QPRP) if for any (non-uniform) QPT oracle algorithm $\mathcal{A}$, P_k with a random $k \leftarrow \mathcal{K}$ is indistinguishable from a truly random permutation π in the sense that*

$$\left|\Pr_k[\mathcal{A}^{P_k,P_k^{-1}}(1^n) = 1] - \Pr_\pi[\mathcal{A}^{\pi,\pi^{-1}}(1^n) = 1]\right| = \mathrm{negl}(n).$$

In addition, P_k is polynomial-time computable on a classical computer.

2.4 Almost Invariance Under Haar Unitaries

The following definition, claim and lemma are taken from [2].

Definition 2.4. *Let $n, q, \ell \in \mathbb{N}$. An $(\ell + n\cdot q)$-qubit state ρ is ϵ-almost invariant under q-fold Haar unitary if*

$$\mathrm{TD}\left(\rho, \mathop{\mathbb{E}}_{U\leftarrow Haar_n}\left[(I_\ell \otimes U^{\otimes q})\rho(I_\ell \otimes (U^\dagger)^{\otimes q})\right]\right) \le \epsilon \tag{4}$$

Claim 2.5. *Let $n, q, \ell \in \mathbb{N}$. Suppose Φ is a quantum channel that is a probabilistic mixture of unitaries on $(\ell + n\cdot q)$ qubits. More precisely, $\Phi(\rho) := \mathbb{E}_{k\leftarrow\mathcal{D}}\left[(I_\ell \otimes V_k^{\otimes q})\rho(I_\ell \otimes (V_k^\dagger)^{\otimes q})\right]$ where $\mathcal{D}$ a distribution over $\{0,1\}^*$, and $V_k : \mathbb{C}^{2^n} \to \mathbb{C}^{2^n}$ is a unitary for every k in the support of D.*

Suppose for a $(\ell + n\cdot q)$-qubit state ρ, $\Phi(\rho)$ is ϵ-almost invariant under q-fold Haar unitary, where ϵ is a negligible function, then the following holds:

$$\mathrm{TD}\left(\Phi(\rho), \mathop{\mathbb{E}}_{U\leftarrow Haar_n}\left[(I_\ell \otimes U^{\otimes q})\rho(I_\ell \otimes (U^\dagger)^{\otimes q})\right]\right) \le \epsilon \tag{5}$$

Lemma 2.6. *Let $n, s, t \in \mathbb{N}$, and define*

$$\rho_{\mathsf{uni}_{s,t}} := \frac{1}{N^{\underline{st}}} \sum_{\substack{z\in\mathcal{U}^\star_{n,st}\\ \sigma\in S^s_t}} |z\rangle\langle\sigma(z)| \ , \tag{6}$$

then for any ℓ qubit state ρ_ℓ, $\rho_\ell \otimes \rho_{\mathsf{uni}_{s,t}}$ is $O(s^2t^2/2^n)$-almost invariant under st-fold Haar unitary (I_ℓ being applied on ρ_ℓ).

This state is close to the output of applying a Haar random unitary on s different orthogonal vectors with t copies of each.

3 Somewhat Pseudorandom Unitaries

Definition 3.1 (Non-adaptive Orthogonal-Inputs Pseudorandom Unitary). *We say that* $\{Gen_n\}_{n\in\mathbb{N}}$ *is a* non-adaptive orthogonal-inputs secure pseudorandom unitary family *if there exists a polynomial* κ *such that:*

- *For every* $k \in \{0,1\}^{\kappa(n)}$, $U_k := Gen_n(k)$ *is a QPT algorithm implementing a unitary operation on* n *qubits.*
- *Fix* $s := s(n), t := t(n)$ *polynomials in* n. *Let* A *be a set and let* $\left\{|\psi^{(1,a)}\rangle, \dots |\psi^{(s,a)}\rangle\right\}$ *be orthogonal states and* ρ_a *be any* ℓ*-qubit state for all* $a \in A$. *Let* p_a *be probabilities such that* $\sum_{a\ inA} p_a = 1$. *There exists a sufficiently large* $n \in \mathbb{N}$, *such that for any (non-uniform) QPT distinguisher* $\mathcal{A}$ *that makes queries of the form*

$$\rho_{in} := \sum_{a\in A} p_a \left(\rho_a \otimes \left(\bigotimes_{j\in[s]} (|\psi^{(j,a)}\rangle\langle\psi^{(j,a)}|)^{\otimes t}\right)\right) \tag{7}$$

to the st*-tensor of the unitary* U_k *it holds that*

$$\left| \Pr_{k\leftarrow\{0,1\}^\kappa}\left[\mathcal{A}\left((I_\ell \otimes U_k^{\otimes st})\rho_{in}(I_\ell \otimes U_k^{\dagger\otimes st})\right) = 1\right] - \Pr_{U\leftarrow Haar_n}\left[\mathcal{A}\left((I_\ell \otimes U^{\otimes st})\rho_{in}(I_\ell \otimes U^{\dagger\otimes st})\right) = 1\right]\right| \leq \mathrm{negl}(n) \tag{8}$$

Theorem 3.2. *Assuming the existence of quantum secure one way functions, there exists a family of non-adaptive orthogonal-inputs secure pseudorandom* real *unitary.*

From Definition 2.4, Claim 2.5, and Lemma 2.6, our goal will be to construct a channel Φ such that its output for copies of orthogonal states looks like almost invariant under st-fold Haar unitary.

The Construction. Let $F, G : \{0,1\}^n \to \{\pm 1\}$ be QPRFs and $P : \{0,1\}^n \to \{0,1\}^n$ be a QPRP. We define the unitary $U_{F,G,P}$ on n qubits as follows:

$$U_{F,G,P} = U_P U_G H^{\otimes n} U_F \ , \tag{9}$$

where

$$U_F = \sum_{x\in\{0,1\}^n} F(x)|x\rangle\langle x|,\ U_G = \sum_{x\in\{0,1\}^n} G(x)|x\rangle\langle x|,\ U_P = \sum_{x\in\{0,1\}^n} |P(x)\rangle\langle x| \ .$$

Invoking Cryptographic Assumptions. We move from pseudorandom F, G, P to truly random f, g, π. The unitary in the random case is denoted by $U_{f,g,\pi}$ and defined similarly.

Claim 3.3. *Let $U_{F,G,P}$ implicitly depend on a key k. Assuming the security of F, G and P, it holds that*

$$\left| \Pr_{k}\left[\mathcal{A}\left(\left(I_\ell \otimes {U_{F,G,P}}^{\otimes st}\right)\rho_{in}\left(I_\ell \otimes {U^\dagger_{F,G,P}}^{\otimes st}\right)\right)=1\right] - \Pr_{f,g,\pi}\left[\mathcal{A}\left(\left(I_\ell \otimes {U_{f,g,\pi}}^{\otimes st}\right)\rho_{in}\left(I_\ell \otimes {U^\dagger_{f,g,\pi}}^{\otimes st}\right)\right)=1\right]\right| \leq \mathrm{negl}(n) \ , \quad (10)$$

where $k = (k_F, k_G, k_P)$ is the key for the PRFs and PRP.

Proof. We define four hybrids. In the first one we query $U_{F,G,P}$, in the second $U_{f,G,P}$, in the third $U_{f,g,P}$, and in the forth $U_{f,g,\pi}$ (all defined similarly to $U_{F,G,P}$). Each two consecutive hybrids are indistinguishable by the security of function replaced between the two hybrids.

4 Analysis

We now turn to analyze the application of $U_{f,g,\pi}$ information theoretically. The main lemma is:

Lemma 4.1. *Let s, t be polynomials in n and let $\{|\psi^{(j)}\rangle\}_{j\in[s]}$ be orthogonal quantum states. Then*

$$\left\| \mathop{\mathbb{E}}_{f,g,\pi}\left[{U_{f,g,\pi}}^{\otimes st}\left(\bigotimes_{j\in[s]}(|\psi^{(j)}\rangle\langle\psi^{(j)}|)^{\otimes t}\right){U^\dagger_{f,g,\pi}}^{\otimes st}\right] - \rho_{\mathsf{uni}_{s,t}}\right\|_1 \leq O(s^6t^4n^2/N) \ . \quad (11)$$

Were the expectation is over sampling random functions f, g and a random permutation π.

4.1 Achieving Flatness

The first two steps in the construction, namely adding a random binary phase with U_f and performing $H^{\otimes n}$, achieve the goal of flattening the state with respect to the standard basis.

Definition 4.2. *A quantum state $|\alpha\rangle = \sum_x \alpha_x|x\rangle$, where $|x\rangle$ are the computational basis elements, is ϵ-flat if $\max_x |\alpha_x|^2 \leq \epsilon$.*

Note that this is equivalent to the min-entropy of the computational-basis measurement of $|\alpha\rangle$ having min-entropy at least $\log(1/\epsilon)$.

Lemma 4.3. *Let $|\beta\rangle = \sum_{x\in\{0,1\}^n} \beta_x |x\rangle$ be a quantum state, $c > 0$, and let $f : \{0,1\}^n \to \{\pm 1\}$ be a random function. Then with probability at least $1 - 2\exp\left(-\left(\frac{c}{4} - \ln(2)\right)n\right)$ the state $H^{\otimes n} U_f |\beta\rangle$ is $c \cdot \frac{n}{2^n}$-flat.*

Proof. Denote $|\xi\rangle = H^{\otimes n} U_f |\beta\rangle$. Let $|y\rangle$ be a standard basis element, and look at $\xi_y = \langle y|\xi\rangle$:

$$\xi_y = \langle y|H^{\otimes n} U_f|\beta\rangle = \frac{1}{2^{n/2}} \sum_{x\in\{0,1\}^n} (-1)^{x\cdot y} \langle x|U_f|\beta\rangle \tag{12}$$

$$= \frac{1}{2^{n/2}} \sum_{x\in\{0,1\}^n} (-1)^{x\cdot y} f(x) \langle x|\beta\rangle \tag{13}$$

$$= \frac{1}{2^{n/2}} \sum_{x\in\{0,1\}^n} (-1)^{x\cdot y} f(x) \beta_x \tag{14}$$

We analyze the real and imaginary parts ξ_y separately. Define the random variables Z_x to be $\Re(2^{-n/2}(-1)^{x\cdot y} f(x)\beta_x)$. It follows that $|Z_x| \le 2^{-n/2}|\beta_x|$. Using Hoeffding's inequality we get

$$\Pr\left[|\Re(\xi_y)| \ge \sqrt{\epsilon/2}\right] \le \exp\left(\frac{-2\cdot(\epsilon/2)}{\sum_x |2\cdot 2^{-n/2}\beta_x|^2}\right) = \exp\left(\frac{-\epsilon\cdot 2^n}{4\sum_x |\beta_x|^2}\right) = \exp(-\epsilon\cdot 2^n/4) \tag{15}$$

Where the second to last equality follows from $|\beta\rangle$ being a quantum state and thus a unit vector. We get $\Pr\left[|\Im(\xi_y)| \ge \sqrt{\epsilon/2}\right] \le \exp(-\epsilon\cdot 2^n/4)$ similarly. Using the bound on both the real part and imaginary part we get:

$$\Pr\left[|\xi_y| \ge \sqrt{\epsilon}\right] \le \Pr\left[|\Re(\xi_y)|^2 + |\Im(\xi_y)|^2 \ge \epsilon\right] \tag{16}$$

$$\le \Pr\left[|\Re(\xi_y)|^2 \ge \epsilon/2 \vee |\Im(\xi_y)|^2 \ge \epsilon/2\right] \tag{17}$$

$$\le \Pr\left[|\Re(\xi_y)| \ge \sqrt{\epsilon/2}\right] + \Pr\left[|\Im(\xi_y)| \ge \sqrt{\epsilon/2}\right] \tag{18}$$

$$\le 2\exp\left(-\epsilon\cdot 2^n/4\right) \tag{19}$$

Finally, we use the union bound over the 2^n entries of $|\xi\rangle$ to get that with probability at most $2\exp\left(-\epsilon\cdot 2^n/4\right)\cdot 2^n$ there exists an entry $|\xi_y| \ge \sqrt{\epsilon}$. Taking $\epsilon = c\cdot\frac{n}{2^n}$ completes the lemma.

Using the union bound, an immediate corollary follows:

Corollary 4.4. *Let $\{|\beta^{(j)}\rangle\}_{j\in[s]}$ be quantum states, $c > 0$, and let $f : \{0,1\}^n \to \{\pm 1\}$ be a random function. Then with probability $1 - s\cdot 2\exp\left(-\left(\frac{c}{4} - \ln(2)\right)n\right)$ all states $H^{\otimes n} U_f |\beta^{(j)}\rangle$ are $c\cdot\frac{n}{2^n}$-flat.*

4.2 Getting from Flat States to Random-Looking Ones

We now prove that applying a random binary phase and a random permutation to t copies of s orthogonal flat vectors with is close in trace distance the almost invariant state from Lemma 2.6. We prove the following lemma:

Lemma 4.5. *Let $g : \{0,1\}^n \to \{\pm 1\}$ be a random function and $\pi : \{0,1\}^n \to \{0,1\}^n$ be a random (inner) permutation. Let $\{|\alpha^{(j)}\rangle\}_{j\in[s]}$ be a set of s arbitrary orthogonal ϵ-flat vectors in $\mathbb{C}^{\{0,1\}^n}$. Denote:*

$$\rho \coloneqq \mathbb{E}_{g,\pi}\left[(U_\pi U_g)^{\otimes st}\left(\otimes_{j\in[s]}|\alpha^{(j)}\rangle\langle\alpha^{(j)}|^{\otimes t}\right)(U_g^\dagger U_\pi^\dagger)^{\otimes st}\right] . \tag{20}$$

Then,

$$\|\rho - \rho_{\mathsf{uni}_{s,t}}\|_1 \leq O((st)^2\epsilon + Ns^6t^4\epsilon^2) . \tag{21}$$

Focusing on Unique States. Recall that $g_z = \prod_{j,i} g(z_{j,i})$. We notice that for all $z, z' \in \{0,1\}^{n\cdot st}$ it holds that $\mathbb{E}_g[g_z g^*_{z'}]$ is equal to 1 if and only if the binary type of z and z' (that is, the histogram of the entries modulus 2) are equal. Otherwise, the expectation is equal to 0. Expressing ρ in the standard basis we get

$$\rho = \mathop{\mathbb{E}}_{g,\pi}\left[\sum_{z,z'\in\{0,1\}^{n\cdot st}} g_z g_{z'} \prod_{\substack{j\in[s]\\ i\in[t]}} \alpha^{(j)}_{z(j,i)}\alpha^{*(j)}_{z'(j,i)}|\pi(z)\rangle\langle\pi(z')|\right] \tag{22}$$

$$= \mathop{\mathbb{E}}_{\pi}\left[\sum_{z,z'\in\{0,1\}^{n\cdot st}} \mathop{\mathbb{E}}_{g}[g_z g^*_{z'}] \prod_{\substack{j\in[s]\\ i\in[t]}} \alpha^{(j)}_{z(j,i)}\alpha^{*(j)}_{z'(j,i)}|\pi(z)\rangle\langle\pi(z')|\right] \tag{23}$$

$$= \mathop{\mathbb{E}}_{\pi}\left[\sum_{z\in\{0,1\}^{n\cdot st}} \sum_{\substack{z'\in\{0,1\}^{n\cdot st}\\ \mathsf{bintype}(z')=\mathsf{bintype}(z)}} \prod_{\substack{j\in[s]\\ i\in[t]}} \alpha^{(j)}_{z(j,i)}\alpha^{*(j)}_{z'(j,i)}|\pi(z)\rangle\langle\pi(z')|\right] , \tag{24}$$

where $\mathsf{bintype}(z)$ is the binary type of z.

Let $\Pi^\star \coloneqq \sum_{z\in\mathcal{U}^\star_{n,st}} |\boldsymbol{z}\rangle\langle\boldsymbol{z}|$ be the uniqueness projector, and let

$$\rho^\star \coloneqq \frac{\Pi^\star\rho\Pi^\star}{\mathrm{Tr}[\Pi^\star\rho]} \tag{25}$$

be the unique restrictions of ρ. We show that ρ is close to its unique restriction.

Claim. It holds that

$$\|\rho - \rho^\star\|_1 \leq O\left((st)^2\epsilon\right) . \tag{26}$$

Proof. Notice that $\Pi^\star\rho(I-\Pi^\star) = (I-\Pi^\star)\rho\Pi^\star = 0$, as $|\pi(z)\rangle$ is in the unique restriction if and only if $\langle\sigma(\pi(z))|$ is in the unique restriction too (which occurs if and only if the binary type/histogram has st entries of 1). It follows that $\rho = \Pi^\star\rho\Pi^\star + (I-\Pi^\star)\rho(I-\Pi^\star)$, and as $\Pi^\star\rho\Pi^\star$ and $(I-\Pi^\star)\rho(I-\Pi^\star)$ are positive semi-definite, it is enough to show that

$$\mathrm{Tr}[(I-\Pi^\star)\rho(I-\Pi^\star)] \leq (st)^2 \cdot \epsilon \ . \tag{27}$$

We note that $\Pi^\star$ is invariant under U_g, U_π, therefore

$$\begin{aligned}
&\mathrm{Tr}[(I-\Pi^\star)\rho(I-\Pi^\star)] \\
&\quad = \mathrm{Tr}\left[(I-\Pi^\star)\mathbb{E}_{g,\pi}\left[(U_\pi U_g)^{\otimes st}\left(\otimes_{j\in[s]}|\alpha^{(j)}\rangle\langle\alpha^{(j)}|^{\otimes t}\right)(U_g^\dagger U_\pi^\dagger)^{\otimes st}\right]\right] &(28)\\
&\quad = \mathbb{E}_{g,\pi}\left[\mathrm{Tr}\left[(U_\pi U_g)^{\otimes st}(I-\Pi^\star)\left(\otimes_{j\in[s]}|\alpha^{(j)}\rangle\langle\alpha^{(j)}|^{\otimes t}\right)(U_g^\dagger U_\pi^\dagger)^{\otimes st}\right]\right] &(29)\\
&\quad = \mathbb{E}_{g,\pi}\left[\mathrm{Tr}\left[(I-\Pi^\star)\left(\otimes_{j\in[s]}|\alpha^{(j)}\rangle\langle\alpha^{(j)}|^{\otimes t}\right)\right]\right] &(30)\\
&\quad = \mathrm{Tr}\left[(I-\Pi^\star)\left(\otimes_{j\in[s]}|\alpha^{(j)}\rangle\langle\alpha^{(j)}|^{\otimes t}\right)\right] \ , &(31)
\end{aligned}$$

which is exactly the probability of measuring $\otimes_{j\in[s]}|\alpha^{(j)}\rangle\langle\alpha^{(j)}|^{\otimes t}$ in the computational basis, and obtaining an (st)-tuple that contains a repetition (i.e. an element in $\{0,1\}^n$ that appears more than once). Due to ϵ-flatness, the probability for this is bounded by $(st)^2 \cdot \epsilon$. ■

Recall that $\mathcal{U}^\star_{n,st}$ is the set of all st length vectors with unique entries from $\{0,1\}^n$. From the definitions of U_g, U_π and Claim 4.2, for $c_1 = \mathrm{Tr}[\Pi^\star\rho_{g,\pi}] \geq \frac{1}{1-\epsilon(st)^2}$ we get

$$\rho^\star = c_1 \cdot \mathop{\mathbb{E}}_{\pi}\left[\sum_{z\in\mathcal{U}^\star_{n,st}}\sum_{\sigma\in S_{st}}\prod_{\substack{j\in[s]\\ i\in[t]}}\alpha^{(j)}_{z(j,i)}\alpha^{*(j)}_{\sigma(z)(j,i)}|\pi(z)\rangle\langle\sigma(\pi(z))|\right] \ . \tag{32}$$

Notice the sum over z' changed to sum over $\sigma \in S_{st}$ (an outer permutation which permutes the positions of the entries) as for z with unique entries it holds that $bintype(z) = bintype(z')$ if and only if there exists $\sigma \in S_{st}$ s.t. $z' = \sigma(z)$.

For all $(j,i) \in [s]\times[t]$ we define $\alpha^{(j,i)} = \alpha^{(j)}$ (since we implicitly think of the index (j,i) as pointing to the j'th qubit group which consists of a t-tensor of $|\alpha^{(j)}\rangle\langle\alpha^{(j)}|$). Changing the order of summation by π^{-1}. We get

$$\rho^\star = c_1 \cdot \sum_{\sigma\in S_{st}}\sum_{z\in\mathcal{U}^\star_{n,st}} \underbrace{\mathbb{E}_{\pi^{-1}}\left[\prod_{\substack{j\in[s]\\ i\in[t]}}\alpha^{(j,i)}_{\pi^{-1}(z)(j,i)}\alpha^{*(j,i)}_{\sigma(\pi^{-1}(z))(j,i)}\right]}_{\text{denote } \nu_{\sigma,z}} |z\rangle\langle\sigma(z)| \ . \tag{33}$$

As π^{-1} is also a random permutation, we get that $\nu_{\sigma,z}$ is independent of z, i.e. $\nu_{\sigma,z} = \nu_\sigma$ for all $z \in \mathcal{U}^\star_{n,st}$. Making a change of variables $x = \pi^{-1}(z)$,

$$\nu_\sigma = \mathbb{E}_{x \in \mathcal{U}^\star_{n,st}} \left[\prod_{\substack{j \in [s] \\ i \in [t]}} \alpha^{(j,i)}_{x(j,i)} \alpha^{*(j,i)}_{\sigma(x)(j,i)} \right] \tag{34}$$

and

$$\rho^\star = c_1 \cdot \sum_{\sigma \in S_{st}} \nu_\sigma \sum_{z \in \mathcal{U}^\star_{n,st}} |z\rangle\langle\sigma(z)| \ . \tag{35}$$

Using Orthogonality to Reach Closeness to an Almost Invariant State. We consider the operator

$$A = \sum_{\sigma \in S_{st}} \nu_\sigma \sum_{z \in \mathcal{U}^\star_{n,st}} |z\rangle\langle\sigma(z)| \ , \tag{36}$$

and show that it is close in trace norm to to the same operator summing only over $\sigma \in S^s_t$. For that, we define the following.

Definition 4.6. *For any permutation $\sigma \in S_{st}$, we consider the associated directed graph G_σ, whose vertex set is $[s] \times [t]$, and there is an edge $(j,i) \to (j',i')$ if and only if $\sigma((j,i)) = (j',i')$. For all $j \in [s]$, we define the j-th vertex-block as the set $\{(j,i)\}_{i \in [t]}$. We sometimes completely associate σ with G_σ.*

We associate each vertex with its outgoing edge. For any vertex $v = (j,i) \in G_\sigma$ with the outgoing edge $(j',i') = \sigma((i,j))$, we denote $j_v = j$, $j'_v = j'$, namely the block-source and block-destination of v in the graph. We say that v is a crossing vertex *if $j_v \neq j'_v$, and otherwise v is* non-crossing*. We denote the set of crossing vertices by cv_σ, and will often omit the subscript when σ is clear from the context. Likewise, we denote the set of non-crossing vertices by $\overline{\mathsf{cv}_\sigma}$.*

The block edge pattern *of σ is the vector $\mathsf{p}_\sigma \in \mathbb{N}^{[s]\times[s]}$, where $\mathsf{p}_\sigma[j,j']$ is the number of crossing vertices from block j to block j'. We say that two permutations are congruent (with respect to S^s_t) if they have the same block edge pattern. It follows that σ, σ' are congruent, denoted $\sigma \cong \sigma'$ if and only if there exist $\sigma_1, \sigma_2 \in S^s_t$ s.t. $\sigma' = \sigma_1 \sigma \sigma_2$. We overload the notation and use p also to denote the congruence class corresponding to this pattern.*

The number of crossing and non-crossing vertices is thus $|\mathsf{cv}|$, and $|\overline{\mathsf{cv}}|$ respectively (so, $|\mathsf{cv}| + |\overline{\mathsf{cv}}| = st$). We note that $|\mathsf{cv}|$, $|\overline{\mathsf{cv}}|$ only depend on the congruence class p.

Under the above definition, and denoting $x(v) = x(j,i)$, we have that

$$\nu_\sigma = \mathbb{E}_{x\in\mathcal{U}^\star_{n,st}}\left[\prod_{\substack{j\in[s]\\ i\in[t]}} \alpha^{(j,i)}_{x(j,i)}\alpha^{*(j,i)}_{\sigma(x)(j,i)}\right] = \mathbb{E}_{x\in\mathcal{U}^\star_{n,st}}\left[\prod_{\substack{j\in[s]\\ i\in[t]}} \alpha^{(j,i)}_{x(j,i)}\prod_{\substack{j\in[s]\\ i\in[t]}}\alpha^{*(j,i)}_{x(\sigma^{-1}(j,i))}\right] \tag{37}$$

$$= \mathbb{E}_{x\in\mathcal{U}^\star_{n,st}}\left[\prod_{\substack{j\in[s]\\ i\in[t]}} \alpha^{(j,i)}_{x(j,i)}\prod_{\substack{j\in[s]\\ i\in[t]}}\alpha^{*\sigma((j,i))}_{x(j,i)}\right] = \mathbb{E}_{x\in\mathcal{U}^\star_{n,st}}\left[\prod_{\substack{j\in[s]\\ i\in[t]}} \alpha^{(j_v)}_{x(v)}\alpha^{*(j'_v)}_{x(v)}\right] \tag{38}$$

$$= \mathbb{E}_{x\in\mathcal{U}^\star_{n,st}}\left[\prod_{v\in\overline{\mathsf{cv}}}\left|\alpha^{(j_v)}_{x(v)}\right|^2\prod_{v\in\mathsf{cv}}\alpha^{(j_v)}_{x(v)}\alpha^{*(j'_v)}_{x(v)}\right] \ . \tag{39}$$

Proposition 4.7. *If $\sigma \cong \sigma'$ then $\nu_\sigma = \nu_{\sigma'}$.*

Proof. Consider a permutation σ. Let us break the operand in the expectation in Eq. (39) into blocks. Namely, for a fixed j consider

$$\prod_{\substack{v\in\overline{\mathsf{cv}}\\ j_v=j}}\left|\alpha^{(j_v)}_{x(v)}\right|^2\prod_{\substack{v\in\mathsf{cv}\\ j_v=j}}\alpha^{(j_v)}_{x(v)}\alpha^{*(j'_v)}_{x(v)} \ . \tag{40}$$

We notice that the expression above only depends on the number and block identities of the neighbors of the elements in the j'th block. Multiplying over all blocks we get ν_σ which remains invariant under (outer) permutations in S^s_t.

We can therefore denote ν_{p} which is the value corresponding to ν_σ for all $\sigma \in \mathsf{p}$. Define

$$A_{\mathsf{p}} \coloneqq \sum_{z\in\mathcal{U}^\star_{n,st}}\sum_{\sigma\in\mathsf{p}}|z\rangle\langle\sigma(z)| = \sum_{z\in\mathcal{U}^\star_{n,st}}|z\rangle\sum_{\sigma\in\mathsf{p}}\langle\sigma(z)| \ . \tag{41}$$

Corollary 4.8. *It holds that*

$$A = \sum_{\mathsf{p}}\nu_{\mathsf{p}}\sum_{z\in\mathcal{U}^\star_{n,st}}\sum_{\sigma\in\mathsf{p}}|z\rangle\langle\sigma(z)| = \sum_{\mathsf{p}}\nu_{\mathsf{p}}A_{\mathsf{p}} \ , \tag{42}$$

We separate the analysis to bounding the norm of A_{p} according to the crossing number of p, counting the number of congruence classes with a certain crossing number, and bounding ν_{p} according to the crossing number of p.

Lemma 4.9. *Let p be with $|\mathsf{cv}| = k$. It holds that*

$$\|A_{\mathsf{p}}\|_1 \leq N^{\underline{st}}\cdot t^k \tag{43}$$

Proof. Partition the space $z \in \mathcal{U}^{\star}_{n,st}$ into parts that are invariant under "block-permutations", i.e. under S_t^s. By definition, each such set contains $(t!)^s$ different z values (recall that all $z(j,i)$ are unique), and the number of partitions is $\frac{N^{\underline{st}}}{(t!)^s}$.

For each partition P, define

$$A_{\mathsf{p},P} = \sum_{z \in P} |z\rangle \sum_{\sigma \in \mathsf{p}} \langle \sigma(z)| \ . \tag{44}$$

For all $z \in P$, the vector $\sum_{\sigma \in \mathsf{p}} \langle \sigma(z)|$ is the same, since the elements of P all differ by a $\tilde{\sigma} \in S_t^s$ permutation, and $\mathsf{p} = \mathsf{p}\tilde{\sigma}$. Thus, $A_{\mathsf{p},P}$ is a rank-1 matrix, and the norm $\|A_{\mathsf{p},P}\|_1$ is the product of the Euclidean norm of the two vectors. In our case,

$$\|A_{\mathsf{p},P}\|_1 = \sqrt{|P|} \cdot \sqrt{|\mathsf{p}|} \tag{45}$$

$$= (t!)^{s/2} \cdot \sqrt{|\mathsf{p}|} \ . \tag{46}$$

Therefore, by the triangle inequality we have that

$$\|A_{\mathsf{p}}\|_1 \leq \frac{N^{\underline{st}}}{(t!)^s} \cdot (t!)^{s/2} \cdot \sqrt{|\mathsf{p}|} \ , \tag{47}$$

Next, we bound the cardinality of p when interpreted as a congruence class (namely, the number of permutations that have edge pattern p):

$$|\mathsf{p}| \leq (t!)^s \cdot t^{2k} \ . \tag{48}$$

We show this by over-counting the set of graphs with a given edge pattern and $|\mathsf{cv}| = k$. We first consider all of the crossing edges, of which there are k by definition. For each such edge, the edge pattern already specifies its source and destination blocks, so we need to choose its specific source and origin vertices within the blocks. There are at most t^2 options for each edge, and thus at most t^{2k} options in general. Then, for all of the other edges, it just remains to go over each of the s blocks of vertices in the graph, and organize the internal edges in the block. We note that any such arrangement can be completed into a permutation in S_t, by orienting the outgoing and incoming edges of the block towards each other arbitrarily. Therefore, the number of arrangements in each block is at most $|S_t| = t!$. It follows that across all blocks, the total number of internal arrangements in bounded by $(t!)^s$. The lemma follows from Eqs. 47 and 48.

Lemma 4.10. *Denote*

$$\mathsf{p}_k = \{\mathsf{p} : |\mathsf{cv}| = k\} \tag{49}$$

the number of congruence classes with crossing number k. Then $|\mathsf{p}_k| \leq s^{2k}$.

Proof. We over-count the elements in p_k. Each edge should be assigned to one of $\binom{s}{2} \leq s^2$ pairs of origin and destination blocks. Therefore, the total number of edge patterns is at most s^{2k}.

We now bound

Lemma 4.11. *Let* p *be with crossing numeber* $|\mathsf{cv}|$*. It holds that*

$$|\nu_{\mathsf{p}}| \leq (N^{\underline{st}})^{-1}((st)^2\epsilon^2 N)^{|\mathsf{cv}|/2} \ . \tag{50}$$

Proof. Consider some $\sigma \in \mathsf{p}$, and recall that

$$\nu_\sigma = \mathbb{E}_{x\in\mathcal{U}^\star_{n,st}}\left[\prod_{v\in\overline{\mathsf{cv}}}\left|\alpha^{(j_v)}_{x(v)}\right|^2\prod_{v\in\mathsf{cv}}\alpha^{(j_v)}_{x(v)}\alpha^{*(j'_v)}_{x(v)}\right] \tag{51}$$

$$= (N^{\underline{st}})^{-1}\sum_{x\in\mathcal{U}^\star_{n,st}}\left[\prod_{v\in\overline{\mathsf{cv}}}\left|\alpha^{(j_v)}_{x(v)}\right|^2\prod_{v\in\mathsf{cv}}\alpha^{(j_v)}_{x(v)}\alpha^{*(j'_v)}_{x(v)}\right] \ . \tag{52}$$

It is possible to sum over the x elements as follows. Go over all v at arbitrary order, and for each v, let $x(v)$ run over all elements in $\{0,1\}^n$ that were not selected in the previous v's, for each such value of $x(v)$ continue to pick value for the next v in the order.

In order to analyze this expression, we consider a more general expression as follows:

$$\tau = \sum_{y_1,\ldots,y_m}\prod_{i\in[m]}\left|\alpha^{(j_i)}_{y_i}\right|^2\delta(\alpha_{y_1},\ldots,\alpha_{y_m})\sum_{z_1}M_1(\alpha_{z_1})\cdots\sum_{z_\ell}M_\ell(\alpha_{z_\ell}) \ , \tag{53}$$

where the indices y_i and z_i run over all of $\{0,1\}^n$ except for the preceding values of the indices. That is, the $y_1,\ldots,y_m$ values are first chosen to be distinct, and then each z_i is chosen in order to be distinct from $y_1,\ldots,y_m$ and all preceding z_i.

The function $\delta(\cdot)$ can be an arbitrary polynomial, and the functions M_i are monomials, where δ and M_i can act on the set of their operands and their complex conjugates. We only consider setting where the total degree of M_i is even. We note that we can always reorder the z_i's without effecting the total value of the expression (maintaining the convention that "later" z_i's take all values except those of y_i's and "previous" z_i's and).

We let d_i denote the total degree of M_i. We say that an index z_i is *loaded* if $d_i \geq 4$, otherwise we say that it is *free* (by our convention this means that $d_i = 2$). As a convention, we always order the summation so that the loaded indices are enumerated on before the free ones. We let ℓ' denote the number of loaded indices. Furthermore, in our setting, any free term i is going to be of the form $M_i(\alpha_{z_i}) = \alpha^{(j)}_{z_i}\alpha^{*(j')}_{z_i}$, or its complex conjugate, for some $j \neq j'$.

We define the *magnitude* of τ as follows, letting δ_0 be the maximal value of δ over all possible inputs that it can take:

$$\mathsf{Mag}(\tau) = \delta_0\prod_{i\in[\ell']}(\epsilon^{d_i/2}N) \tag{54}$$

$$= \delta_0 N^{\ell'}\epsilon^{(\sum_i d_i)/2} \ . \tag{55}$$

We let $d = \sum_{i\in[\ell']} d_i$ denote the total degree of all loaded elements.

We now recall that $|\alpha^{(j)}\rangle, |\alpha^{(j')}\rangle$ are orthogonal for $j \neq j'$, and therefore $\sum_{z\in\{0,1\}^n} \alpha_z^{(j)} \alpha^{*(j')}_z = 0$. It therefore follows that if τ has any free indices, i.e. by our convention if its ℓ index is free, then we have that (up to complex conjugation)

$$\sum_{z_\ell} M_\ell(\alpha_{z_\ell}) = \sum_{z_\ell} \alpha^{(j)}_{z_\ell} \alpha^{*(j')}_{z_\ell} = 0 - \sum_{i<\ell} \alpha^{(j)}_{z_i} \alpha^{*(j')}_{z_i} + \delta_\ell(\alpha_{y_1}, \ldots \alpha_{y_m}) \ . \tag{56}$$

where $|\delta_\ell| \leq \epsilon \cdot (st)$ from ϵ flatness.

Each term of the form $\alpha^{(j)}_{z_i} \alpha^{*(j')}_{z_i}$ now "joins" M_i and so it either creates a new loaded term, if z_i was not previously loaded, decreasing the value of Mag by $\epsilon^2 N$, or adds 2 to the degree of a pre-existing loaded term if z_i was previously loaded, decreasing the value of Mag by ϵ. Furthermore, multiplying by δ_ℓ would decrease the value of the "global" delta by a factor of $\epsilon \cdot (st)$. Therefore, we can write τ as a sum of ℓ terms, each of which conforms with the general form of Eq. (53), but with only $\ell - 1$ indices (rather than ℓ), namely:

$$\tau = \sum_{i\in[\ell]} \tau_i \ , \tag{57}$$

and it holds that

$$\mathsf{Mag}(\tau_i) \leq \max\{\epsilon^2 \, N, \epsilon, \epsilon \cdot (st)\} \mathsf{Mag}(\tau) \leq (\epsilon^2 \, N(st)) \cdot \mathsf{Mag}(\tau) \ . \tag{58}$$

We can now prove the following inductive claim:

Claim. Let τ be with parameters m, ℓ, ℓ' as above, and assume $(\epsilon^2 N(st)\ell) < 1$ then it holds that

$$|\tau| \leq (\epsilon^2 \, N(st)\ell)^{\frac{\ell-\ell'}{2}} \cdot \mathsf{Mag}(\tau) \ . \tag{59}$$

Proof. We prove this inductively over the value of $\ell - \ell'$. For the base case, consider the setting where $\ell' = \ell$. Notice that $\sum_{y_1,\ldots,y_m} \prod_{i\in[m]} \left|\alpha^{(j_i)}_{y_i}\right|^2 \leq 1$. Therefore, τ is a (sub) convex combination of elements of the form

$$\delta(\alpha_{y_1}, \ldots, \alpha_{y_m}) \sum_{z_1} M_1(\alpha_{z_1}) \cdots \sum_{z_\ell} M_\ell(\alpha_{z_\ell}) \ ,$$

that are each bounded in absolute value by $\delta_0 N^{\ell'} \epsilon^{d/2}$, which we show below. It follows that if τ has no free terms, then $|\tau| \leq \mathsf{Mag}(\tau)$, where $\mathsf{Mag}(\tau)$ is given by Eq. (54).

Indeed, each such element is a product of δ, times a product of ℓ' loaded sums. The total number of summands over all the sums is at most $N^{\ell'}$ (as $l' = l$). Each element in the sum is a product of d_i elements from α. By flatness, each element of α has absolute value at most $\sqrt{\epsilon}$, and therefore each element in the sum has absolute value at most $\epsilon^{d_i/2}$. We get a value that is bounded by $\delta_0 N^{\ell'} \epsilon^{d/2}$ as required.

Now consider the case where $\ell > \ell'$. In this case, we can write $\tau = \sum_{i\in[\ell]} \tau_i$ as above. We notice that for each τ_i, we have $\ell_i = \ell - 1$, and $\ell'_i \leq \ell' + 1$, so $\ell - \ell'$ shrinks by at most 2. Therefore we get the bound

$$|\tau| \leq \sum_{i\in[\ell]} |\tau_i| \tag{60}$$

$$\leq \sum_i (\epsilon^2 N(st)\ell_i)^{\frac{\ell_i - \ell'_i}{2}} \cdot \mathsf{Mag}(\tau_i) \qquad \text{(induction)} \tag{61}$$

$$\leq \sum_i (\epsilon^2 N(st)\ell)^{\frac{\ell-\ell'}{2}-1} \cdot \mathsf{Mag}(\tau_i) \qquad (l_i \leq l, \ell_i - \ell'_i \geq \ell - \ell' - 2) \tag{62}$$

$$\leq \ell \cdot (\epsilon^2 N(st)\ell)^{\frac{\ell-\ell'}{2}-1} \cdot (\epsilon^2 N(st)) \cdot \mathsf{Mag}(\tau) \qquad \text{(Eq. (58))} \tag{63}$$

$$\leq (\epsilon^2 N(st)\ell)^{\frac{\ell-\ell'}{2}} \cdot \mathsf{Mag}(\tau) \tag{64}$$

and the claim thus follows. ■

Now, let us go back to our expression for ν_σ. We can write it as $\nu_\sigma = (N^{\underline{st}})^{-1}\tau$, where τ has the form as above, with $\delta = 1$, $\ell = |\mathsf{cv}| \leq st$, and $\ell' = 0$, thus $\mathsf{Mag}(\tau) = 1$. Claim 4.2 therefore guarantees that

$$|\tau| \leq (\epsilon^2 N(st)^2)^{|\mathsf{cv}|/2} , \tag{65}$$

and the bound for ν_σ thus follows.

Corollary 4.12. *Let $\gamma = \epsilon^2 N(st)^2$, and assume $t^2 s^4 \gamma < 1/4$, then it holds that*

$$\sum_{\mathsf{p}:|\mathsf{cv}|>0} \|\nu_{\mathsf{p}} A_{\mathsf{p}}\|_1 \leq 2t^2 s^4 \gamma = 2t^4 s^6 \epsilon^2 N \tag{66}$$

Proof. We derive the bound in the following equation. We note that $|\mathsf{cv}|$ cannot be equal to 1 since for every outgoing edge from a block there needs to be an incoming edge. We also recall the notation of $\mathsf{p}_k = \{\mathsf{p} : |\mathsf{cv}| = k\}$, introduced in Lemma 4.10 (namely, p_k is a set whose elements are congruence classes p).

$$\sum_{\mathsf{p}:|\mathsf{cv}|>0} \|\nu_{\mathsf{p}} A_{\mathsf{p}}\|_1 = \sum_{k=2}^{st} \sum_{\mathsf{p}\in\mathsf{p}_k} |\nu_{\mathsf{p}}| \, \|A_{\mathsf{p}}\|_1 \tag{67}$$

$$\leq \sum_{k=2}^{st} \sum_{\mathsf{p}\in\mathsf{p}_k} \gamma^{k/2} t^k \qquad \text{Lemmas 4.11 and 4.9} \tag{68}$$

$$\leq \sum_{k=2}^{st} \gamma^{k/2} t^k s^{2k} \qquad \text{Lemma 4.10} \tag{69}$$

$$= \sum_{k=2}^{st} (ts^2\sqrt{\gamma})^k \tag{70}$$

$$\leq \frac{(ts^2\sqrt{\gamma})^2}{1 - ts^2\sqrt{\gamma}} \tag{71}$$

$$\leq 2t^2 \, s^4 \gamma \tag{72}$$

and the lemma follows.

We can now prove Lemma 4.5

Proof (Proof of Lemma 4.5). Using the triangle inequality,

$$\|\rho - \rho_{\mathsf{uni}_{s,t}}\|_1 \leq \|\rho - \rho^\star\|_1 + \left\| \rho^\star - c_1 \cdot \sum_{\mathsf{p}:|\mathsf{cv}|=0} \nu_{\mathsf{p}} A_{\mathsf{p}} \right\|_1 + \left\| c_1 \cdot \sum_{\mathsf{p}:|\mathsf{cv}|=0} \nu_{\mathsf{p}} A_{\mathsf{p}} - \rho_{\mathsf{uni}_{s,t}} \right\|_1 . \tag{73}$$

We bound each term separately.

$$\|\rho^\star - c_1 \cdot \sum_{\mathsf{p}:|\mathsf{cv}|=0} \nu_{\mathsf{p}} A_{\mathsf{p}}\|_1 = \|c_1 \cdot \sum_{\mathsf{p}:|\mathsf{cv}|>0} \nu_{\mathsf{p}} A_{\mathsf{p}}\|_1 \tag{74}$$

$$\leq c_1 \cdot \sum_{\mathsf{p}:|\mathsf{cv}|>0} \|\nu_{\mathsf{p}} A_{\mathsf{p}}\|_1 \tag{75}$$

$$\leq c_1 \cdot 2t^2 \, s^4 \gamma \tag{76}$$

$$= c_1 \cdot 2\epsilon^2 \, N s^6 t^4 \tag{77}$$

We note that there is one congruence class with zero crossing, which contains the permutations $\sigma \in S_t^s$. Denote this congruence class by p_0, so $\sum_{\mathsf{p}:|\mathsf{cv}|=0} A_{\mathsf{p}} = A_{\mathsf{p}_0}$. Recall that

$$\rho_{\mathsf{uni}_{s,t}} = \frac{1}{N^{\underline{st}}} \sum_{\substack{z\in\mathcal{U}^\star_{n,st} \\ \sigma\in S_t^s}} |z\rangle\langle\sigma(z)| = \frac{1}{N^{\underline{st}}} A_{\mathsf{p}_0} \tag{78}$$

As $\mathrm{Tr}(\rho^\star) = 1$, we have

$$\left|1 - c_1 \cdot \nu_{\mathsf{p}_0} N^{\underline{st}}\right| = \left|1 - \mathrm{Tr}\left(c_1 \cdot \nu_{\mathsf{p}_0} A_{\mathsf{p}_0}\right)\right| \leq c_1 \cdot 2\epsilon^2 \, N s^6 t^4 \, . \tag{79}$$

It follows that

$$\frac{1 - c_1 \cdot 2\epsilon^2 \, N s^6 t^4}{c_1} \frac{1}{N^{\underline{st}}} \leq \nu_{\mathsf{p}_0} \leq \frac{1 + c_1 \cdot 2\epsilon^2 \, N s^6 t^4}{c_1} \frac{1}{N^{\underline{st}}} \tag{80}$$

and so,

$$\left\| c_1 \cdot \sum_{\mathsf{p}:|\mathsf{cv}|=0} \nu_{\mathsf{p}} A_{\mathsf{p}} - \rho_{\mathsf{uni}_{s,t}} \right\|_1 = \left\| c_1 \cdot \nu_{\mathsf{p}_0} A_{\mathsf{p}_0} - \frac{1}{N^{\underline{st}}} A_{\mathsf{p}_0} \right\|_1 \leq c_1 \cdot 2\epsilon^2 \, N s^6 t^4 \tag{81}$$

and it follows that

$$\|\rho^\star - \rho_{\mathsf{uni}_{s,t}}\|_1 \leq O(\epsilon^2 \, N s^6 t^4) \tag{82}$$

4.3 Proving the Main Lemma and Theorem

We combine the results of Sects. 4.1 and 4.2 to prove the main lemma.

Proof (Proof of Lemma 4.1). Let s, t be polynomials in n and let $\{|\psi^{(j)}\rangle\}_{j\in[s]}$ be orthogonal quantum states. From corollary 4.4, choosing $c = 8$ we get that $\forall_{j\in[s]} H^{\otimes n} U_f |\psi^{(j)}\rangle$ is $\frac{8n}{N}$-flat with probability at least $1 - s \cdot 2\exp\left(-\left(\frac{8}{4} - \ln(2)\right) n\right) \geq 1 - 2^{-n} = 1 - \frac{1}{N}$.

Assuming flatness of these states, we can now use lemma 4.5 and get that:

$$\left\| \mathop{\mathbb{E}}_{f,g,\pi} \left[{U_{f,g,\pi}}^{\otimes st} \left(\bigotimes_{j\in[s]} (|\psi^{(j)}\rangle\langle\psi^{(j)}|)^{\otimes t} \right) {U^\dagger_{f,g,\pi}}^{\otimes st} \right] - \rho_{\mathsf{uni}_{s,t}} \right\|_1 \leq O\left(\frac{(st)^2 8n}{N} + N s^6 t^4 \left(\frac{9n}{N}\right)^2 + \frac{1}{N} \right) = O\left(\frac{s^6 t^4 n^2}{N} \right) . \tag{83}$$

We conclude with a proof for theorem 3.2.

Proof (Proof of Theorem 3.2). Taking $Gen_n(k) = U_k$ to be $U_{F,G,P}$ (where k is split into keys for F, G and P), we get that it is indeed a QPT algorithm on n qubits. We now prove the security requirement.

Recall that ρ_{in} is promised to be of the form

$$\rho_{in} = \sum_{a\in A} p_a \left(\rho_a \otimes \left(\bigotimes_{j\in[s]} (|\psi^{(j,a)}\rangle\langle\psi^{(j,a)}|)^{\otimes t} \right) \right) \tag{84}$$

for orthogonal sets of states $\left\{|\psi^{(1,a)}\rangle, \ldots, |\psi^{(s,a)}\rangle\right\}$. Define the channel Φ to be

$$\Phi(\rho) = \mathop{\mathbb{E}}_{f,g,\pi} \left[\left(I_\ell \otimes {U_{f,g,\pi}}^{\otimes st} \right) \rho \left(I_\ell \otimes {U^\dagger_{f,g,\pi}}^{\otimes st} \right) \right] . \tag{85}$$

Performing Φ on ρ_{in} results in the state

$$\Phi(\rho_{in}) = \sum_{a\in A} p_a \left(\rho_a \otimes \mathop{\mathbb{E}}_{f,g,\pi} \left[{U_{f,g,\pi}}^{\otimes st} \left(\bigotimes_{j\in[s]} (|\psi^{(j,a)}\rangle\langle\psi^{(j,a)}|)^{\otimes t} \right) {U^{\dagger}_{f,g,\pi}}^{\otimes st} \right] \right) . \tag{86}$$

By Lemma 4.1 and the fact that $\sum_{a\in A} p_a = 1$ we get that

$$\left\| \Phi(\rho_{in}) - \sum_{a\in A} p_a(\rho_a \otimes \rho_{\mathsf{uni}_{s,t}}) \right\|_1 \leq O\left(\frac{s^6t^4n^2}{N}\right) . \tag{87}$$

Together with Lemma 2.6, we get by the triangle inequality that $\Phi(\rho_{in})$ is an $O(s^6t^4n^2/N + s^2t^2/N) = O(s^6t^4n^2/N)$ almost invariant state. Using Claim 2.5 we get

$$\mathrm{TD}\left(\Phi(\rho_{in}), \mathop{\mathbb{E}}_{U\leftarrow Haar_n} \left[(I_\ell \otimes U^{\otimes st})\rho_{in}(I_\ell \otimes (U^{\dagger})^{\otimes st})\right]\right) \leq O(s^6t^4n^2/N) . \tag{88}$$

By Claim 3.3 we have

$$\left| \Pr_k \left[\mathcal{A}\left(\left(I_\ell \otimes {U_{F,G,P}}^{\otimes st}\right)\rho_{in}\left(I_\ell \otimes {U^{\dagger}_{F,G,P}}^{\otimes st}\right)\right) = 1\right] - \Pr_{f,g,\pi} \left[\mathcal{A}\left(\left(I_\ell \otimes {U_{f,g,\pi}}^{\otimes st}\right)\rho_{in}\left(I_\ell \otimes {U^{\dagger}_{f,g,\pi}}^{\otimes st}\right)\right) = 1\right] \right| \leq \mathrm{negl}(n) . \tag{89}$$

We finish by combining Eqs. 89 and 88 to get

$$\left| \Pr_k \left[\mathcal{A}\left(\left(I_\ell \otimes {U_{F,G,P}}^{\otimes st}\right)\rho_{in}\left(I_\ell \otimes {U^{\dagger}_{F,G,P}}^{\otimes st}\right)\right) = 1\right] - \Pr_{U\leftarrow Haar_n} \left[\mathcal{A}\left((I_\ell \otimes U^{\otimes st})\rho_{in}(I_\ell \otimes {U^{\dagger}}^{\otimes st})\right) = 1\right] \right| \leq \mathrm{negl}(n) + O(s^6t^4n^2/N) = \mathrm{negl}(n) , \tag{90}$$

as needed to satisfy the security Definition 3.1.

Acknowledgments. Zvika Brakerski and Nir Magrafta are supported by the Israel Science Foundation (Grant No. 3426/21), and by the Horizon Europe Research and Innovation Program via ERC Project ACQUA (Grant 101087742).

We thank Omri Shmueli for multiple contributions to the results presented in this work. We would also like to thank Yanglin Hu and Marco Patrick Tomamichel for pointing out a gap in a proof in the previous version of the manuscript.

References

1. Alagic, G., Majenz, C., Russell, A.: Efficient simulation of random states and random unitaries. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020. LNCS, vol. 12107, pp. 759–787. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-45727-3_26

2. Ananth, P., Gulati, A., Kaleoglu, F., Lin, Y.T.: Pseudorandom isometries. arXiv preprint arXiv:2311.02901 (2023)
3. Ananth, P., Gulati, A., Qian, L., Yuen, H.: Pseudorandom (function-like) quantum state generators: new definitions and applications. In: Proceedings of the Theory of Cryptography Conference (TCC), pp. 237–265. Springer, Cham (2022)
4. Behera, A., Brakerski, Z., Sattath, O., Shmueli, O.: Pseudorandomness with proof of destruction and applications. Cryptology ePrint Archive (2023)
5. Brakerski, Z., Canetti, R., Qian, L.: On the computational hardness needed for quantum cryptography. In: Proceedings of the 14th Innovations in Theoretical Computer Science Conference (ITCS) (2023)
6. Brakerski, Z., Shmueli, O.: (pseudo) random quantum states with binary phase. In: Proceedings of the Theory of Cryptography Conference (TCC), pp. 229–250. Springer, Cham (2019)
7. Brakerski, Z., Shmueli, O.: Scalable pseudorandom quantum states. In: Proceedings of the 40th Annual International Cryptology Conference (CRYPTO), pp. 417–440. Springer, Cham (2020)
8. Chen, C.F., Bouland, A., Brandão, F.G., Docter, J., Hayden, P., Xu, M.: Efficient unitary designs and pseudorandom unitaries from permutations. arXiv preprint arXiv:2404.16751 (2024)
9. Giurgica-Tiron, T., Bouland, A.: Pseudorandomness from subset states. arXiv preprint arXiv:2312.09206 (2023)
10. Goldreich, O., Goldwasser, S., Micali, S.: How to construct random functions. J. ACM **33**(4), 792–807 (1986)
11. Håstad, J., Impagliazzo, R., Levin, L.A., Luby, M.: A pseudorandom generator from any one-way function. SIAM J. Comput. **28**(4), 1364–1396 (1999)
12. Haug, T., Bharti, K., Koh, D.E.: Pseudorandom unitaries are neither real nor sparse nor noise-robust. arXiv preprint arXiv:2306.11677 (2023)
13. Jeronimo, F.G., Magrafta, N., Wu, P.: Subset states and pseudorandom states. arXiv preprint arXiv:2312.15285 (2023)
14. Ji, Z., Liu, Y.K., Song, F.: Pseudorandom quantum states. In: Proceedings of the 38th Annual International Cryptology Conference (CRYPTO), pp. 126–152. Springer, Cham (2018)
15. Lu, C., Qin, M., Song, F., Yao, P., Zhao, M.: Quantum pseudorandom scramblers. arXiv preprint arXiv:2309.08941 (2023)
16. Luby, M., Rackoff, C.: How to construct pseudorandom permutations from pseudorandom functions. SIAM J. Comput. **17**(2), 373–386 (1988)
17. Metger, T., Poremba, A., Sinha, M., Yuen, H.: Simple constructions of linear-depth t-designs and pseudorandom unitaries. arXiv preprint arXiv:2404.12647 (2024)
18. Zhandry, M.: How to construct quantum random functions. In: Proceedings of the IEEE 53rd Annual Symposium on Foundations of Computer Science (FOCS), pp. 679–687. IEEE (2012)
19. Zhandry, M.: A note on quantum-secure prps. arXiv preprint arXiv:1611.05564 (2016)

Split-State Non-malleable Codes and Secret Sharing Schemes for Quantum Messages

Naresh Goud Boddu[1], Vipul Goyal[1], Rahul Jain[2], and João Ribeiro[3,4(✉)]

[1] NTT Research, Sunnyvale, CA, USA
naresh.boddu@ntt-research.com, vipul@vipulgoyal.org
[2] Centre for Quantum Technologies and National University of Singapore, Singapore, Singapore
rahul@comp.nus.edu.sg
[3] Instituto de Telecomunicações, Lisbon, Portugal
[4] Departamento de Matemática, Instituto Superior Técnico, Universidade de Lisboa, Lisbon, Portugal
jribeiro@tecnico.ulisboa.pt

Abstract. Non-malleable codes are fundamental objects at the intersection of cryptography and coding theory. These codes provide security guarantees even in settings where error correction and detection are impossible, and have found applications to several other cryptographic tasks. One of the strongest and most well-studied adversarial tampering models is 2-split-state tampering. Here, a codeword is split into two parts which are stored in physically distant servers, and the adversary can then independently tamper with each part using arbitrary functions. This model can be naturally extended to the secret sharing setting with several parties by having the adversary independently tamper with each share. Previous works on non-malleable coding and secret sharing in the split-state tampering model only considered the encoding of *classical* messages. Furthermore, until recent work by Aggarwal, Boddu, and Jain (IEEE Trans. Inf. Theory 2024 & arXiv 2022), adversaries with quantum capabilities and *shared entanglement* had not been considered, and it is a priori not clear whether previous schemes remain secure in this model.

In this work, we introduce the notions of split-state non-malleable codes and secret sharing schemes for quantum messages secure against quantum adversaries with shared entanglement. Then, we present explicit constructions of such schemes that achieve low-error non-malleability. More precisely, for some constant $c > 0$, we construct efficiently encodable and decodable split-state non-malleable codes and secret sharing schemes for quantum messages preserving entanglement with external systems and achieving security against quantum adversaries having shared entanglement with codeword length n, any message length at most n^c, and error $\varepsilon = 2^{-n^c}$. In the easier setting of *average-*

J. Ribeiro—Work done while at NOVA LINCS and NOVA School of Science and Technology.

E. Boyle and M. Mahmoody (Eds.): TCC 2024, LNCS 15365, pp. 60–93, 2025.
https://doi.org/10.1007/978-3-031-78017-2_3

case non-malleability, we achieve efficient non-malleable coding with rate close to 1/11.

1 Introduction

Non-Malleable Codes (NMCs), introduced in the work of Dziembowski, Pietrzak, and Wichs [37], are now considered fundamental cryptographic primitives that provide security guarantees even in adversarial settings where error correction and detection are impossible. Informally, NMCs guarantee that an adversary cannot change the encoding of a message into that of a related message. They encode a classical message M into a codeword C in such a way that tampering C into $f(C)$ using an allowed tampering function f results in the decoder either outputting the original message M or a message that is unrelated/independent of M. Note that it is impossible to construct NMCs that protect against *arbitrary* tampering functions. This is because an adversary could simply apply the decoder to C, recovering the message M, and then output the encoding of a related message $M + 1$ as the tampered version of C. Therefore, previous work on non-malleable coding has focused on constructing NMCs for restricted, but still large and meaningful, classes of tampering functions.

One of the strongest and most studied tampering models is *split-state tampering*, introduced by Liu and Lysyanskaya [53]. In the 2-state version of this model (which is the hardest), we view a codeword C as being composed of two parts, E_1 and E_2, and an adversary is allowed to *independently* tamper with each part using an arbitrary tampering function. In other words, a split-state tampering adversary consists of a pair of arbitrary functions (f, g), and a codeword (E_1, E_2) is tampered to $(f(E_1), g(E_2))$ (see Fig. 1 for a diagram of this model). The split-state model is meaningful because we can imagine that the two parts of the codeword are stored in different physically isolated servers, making communication between tampering adversaries infeasible.

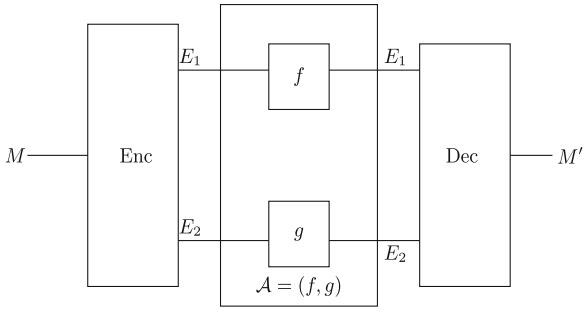

Fig. 1. Classical split-state tampering model.

The notion of non-malleable secret sharing has also been widely studied as a strengthening of non-malleable codes. Secret sharing, dating back to the work of

Blakley [18] and Shamir [56], is a fundamental cryptographic primitive where a dealer encodes a secret into p shares and distributes them among p parties. Each secret sharing scheme has an associated *monotone*[1] set $\Gamma \subseteq 2^{[p]}$, usually called an *access structure*, whereby any set of parties $T \in \Gamma$, called *authorized* sets, can reconstruct the secret from their shares, but any *unauthorized* set of parties $T \notin \Gamma$ gains essentially no information about the secret. One of the most natural and well-studied types of access structures are *threshold* access structures, where a set of parties T is authorized if and only if $|T| \geq t$ for some threshold t.

Non-Malleable Secret Sharing (NMSS), generalizing non-malleable coding, was introduced by Goyal and Kumar [42] and has received significant interest in the past few years in the classical setting. NMSS schemes additionally guarantee that an adversary who is allowed to tamper all the shares (according to some restricted tampering model) cannot make an authorized set of parties reconstruct a different but related secret. Non-malleable secret sharing is particularly well-studied in the context of the split-state tampering model described above, whereby an adversary can independently tamper with each share. Once again, the motivation is that shares are being held in physically distant devices, making communication between the several tampering adversaries infeasible.

The split-state and closely related tampering models for codes and secret sharing schemes have witnessed a flurry of work in the past decade [3,4,6,7,10,21,23,27–30,36,38,42,43,45,49–53], culminating in recent explicit constructions of classical split-state NMCs of rate 1/3 [6] and constructions with a smaller constant rate but also smaller error [52] (with a known upper bound on the rate being 1/2 [30]). Split-state NMCs and NMSS schemes and related notions have also found applications in other cryptographic tasks, such as non-malleable commitments, secure message transmission, and non-malleable signatures [3,27,42–44,57].

Split-State Tampering and Quantum Computing. Given the rapid development of quantum technologies, it is natural to consider NMCs and NMSS schemes in the quantum setting and examine how adversaries with quantum capabilities affect previous assumptions made about tampering models. For instance, all known NMCs are tailored to classical messages, but it is equally important to design non-malleable coding schemes that allow us to encode quantum states as well. Moreover, the possibility of attackers with quantum capabilities challenges the *independence assumption* made in the split-state models described earlier. Although servers holding different parts of the codeword (or different secret shares) may be physically isolated from each other, the tampering adversaries attacking each server may have pre-shared a large amount of entangled quantum states. Access to such shared entanglement can provide non-trivial advantages to these adversaries beyond what has been considered in the classical tampering models. For example, in the Clauser-Horne-Shimony-Holt (CHSH) game [31], non-communicating parties can use local measurements on both halves of an

[1] A set $\Gamma \subseteq 2^{[p]}$ is *monotone* if $A \in \Gamma$ and $A \subseteq B$ imply that $B \in \Gamma$.

EPR state to achieve a higher success probability than what is possible using fully classical strategies. Therefore, it is not clear whether any of the existing classical NMCs and NMSS schemes remain secure in the augmented split-state tampering models where the adversary is allowed to make use of arbitrary shared entanglement across multiple states (see Fig. 2 for a diagram of the split-state tampering model for two states). The only exception to this are the recent work of Aggarwal, Boddu, and Jain [2] and the concurrent work of Batra, Boddu, and Jain [16]. The former constructs explicit NMCs for *classical* messages that are secure in the 2-split-state tampering model, while the latter focuses on explicit quantum-secure *non-malleable randomness encoders* (NMRE) with a higher rate in the same tampering model, and uses these objects to construct non-malleable codes in the 3-split-state model[2] for quantum messages.

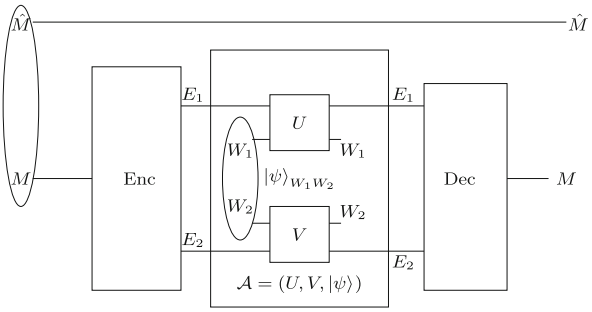

Fig. 2. Split-state tampering model with shared entanglement. This shared entanglement is stored in registers W_1 and W_2.

The shortcomings of existing split-state non-malleable coding schemes in the face of quantum messages and adversaries raise the following natural question:

Can we design efficient 2-split-state NMCs and split-state NMSS schemes for *quantum messages* secure against quantum adversaries with shared entanglement?

We resolve this question in the affirmative.

1.1 Our Contributions

Split-State Non-malleable Codes for Quantum Messages. As our first contribution, we propose a definition of split-state non-malleability for quantum messages against adversaries with shared entanglement. Our definition is a natural extension of the one considered for classical messages in the literature [37].

[2] Meaning that the codeword is divided into three parts and the adversary tampers each part independently. We note that constructing NMCs in the 3-split-state model is considerably easier than in the 2-split-state model.

Here, we present it specifically for the 2-split-state case, which is our main setting of interest.

Let σ_M be an arbitrary state in a message register M, and $\sigma_{M\hat{M}}$ be its canonical purification. We consider (2-split-state) coding schemes given by an encoding Completely Positive Trace-Preserving (CPTP) map $\mathrm{Enc} : \mathcal{L}(\mathcal{H}_M) \to \mathcal{L}(\mathcal{H}_{E_1} \otimes \mathcal{H}_{E_2})$ and a decoding CPTP map $\mathrm{Dec} : \mathcal{L}(\mathcal{H}_{E_1} \otimes \mathcal{H}_{E_2}) \to \mathcal{L}(\mathcal{H}_M)$, where $\mathcal{L}(\mathcal{H})$ is the space of all linear operators in the Hilbert space $\mathcal{H}$.

The most basic property we require of this coding scheme $(\mathrm{Enc}, \mathrm{Dec})$ is correctness (which includes preserving entanglement with external systems), i.e.,

$$\mathrm{Dec}(\mathrm{Enc}(\sigma_{M\hat{M}})) = \sigma_{M\hat{M}},$$

where we use the shorthand T to represent the CPTP map $T \otimes \mathbb{I}$ whenever the action of the identity operator $\mathbb{I}$ is clear from context.

Before we proceed to define split-state non-malleability, we describe the split-state adversarial tampering model in the quantum setting. Let $\rho_{E_1E_2} = \mathrm{Enc}(\sigma_M)$ be the split-state encoding of message σ_M. A split-state tampering adversary $\mathcal{A}$ is specified by two tampering maps[3] $U : \mathcal{L}(\mathcal{H}_{E_1} \otimes \mathcal{H}_{W_1}) \to \mathcal{L}(\mathcal{H}_{E_1} \otimes \mathcal{H}_{W_1})$ and $V : \mathcal{L}(\mathcal{H}_{E_2} \otimes \mathcal{H}_{W_2}) \to \mathcal{L}(\mathcal{H}_{E_2} \otimes \mathcal{H}_{W_2})$ along with a quantum state $|\psi\rangle_{W_1W_2}$ that captures the shared entanglement between the non-communicating tampering adversaries. Finally, the decoding procedure Dec is applied to the tampered codeword. Figure 2 presents a diagram of this tampering model. Let

$$\eta = \mathrm{Dec}\left((U \otimes V)\left(\mathrm{Enc}(\sigma_{M\hat{M}}) \otimes |\psi\rangle\langle\psi|\right)(U^\dagger \otimes V^\dagger)\right)$$

be the final state after applying the split-state tampering adversary $\mathcal{A}$ followed by the decoding procedure.

We are now ready to define split-state non-malleability of the coding scheme $(\mathrm{Enc}, \mathrm{Dec})$.

Definition 1 (Non-Malleable Codes for Quantum Messages). *We say that the coding scheme* $(\mathrm{Enc}, \mathrm{Dec})$ *is a* (worst-case) ε-non-malleable code for quantum messages *if for every split-state adversary* $\mathcal{A} = (U, V, |\psi\rangle_{W_1W_2})$ *and every quantum message* σ_M *(with canonical purification* $\sigma_{M\hat{M}}$*) it holds that*

$$\eta_{M\hat{M}} \approx_\varepsilon p_{\mathcal{A}}\sigma_{M\hat{M}} + (1 - p_{\mathcal{A}})\gamma_M^{\mathcal{A}} \otimes \sigma_{\hat{M}}, \tag{1}$$

where $p_{\mathcal{A}} \in [0, 1]$ *and* $\gamma_M^{\mathcal{A}}$ *depend only*[4] *on the split-state adversary* $\mathcal{A}$*, and* $\approx_\varepsilon$ *denotes that the two states are* ε*-close in trace distance.*

[3] Tampering maps are assumed to be unitary without any loss of generality. This is because, in the presence of unbounded arbitrary shared entanglement, tampering with unitary maps is equivalent to tampering with CPTP maps. More precisely, consider a tampering adversary that uses two CPTP maps Φ_1 and Φ_2 acting on registers E_1W_1 and E_2W_2, respectively. Then, the action of this adversary is equivalent to another adversary who tampers using Stinespring isometry extensions U and V of Φ_1 and Φ_2, respectively, which act on $E_1W_1A_1$ and $E_2W_2A_2$, respectively, where A_1 and A_2 are unentangled ancilla registers set to $|0\rangle$ without loss of generality and can be seen as part of the shared entanglement.

[4] By this, we mean that $p_{\mathcal{A}}$ can be computed and the state $\gamma_M^{\mathcal{A}}$ can be prepared without the knowledge of the input message $\sigma_{M\hat{M}}$.

If Eq. (1) *is only guaranteed to hold when* σ_M *is the maximally mixed state, then we say that* (Enc, Dec) *is an* average-case ε-non-malleable code for quantum messages.

Remark 1. Intuitively, our definition of average-case non-malleability for quantum messages in Definition 1 is analogous to requiring that the average non-malleability error of a given classical code is small when averaged over a uniformly random message. Later, in Lemma 5, we show that every average-case non-malleable code for quantum messages is also a worst-case non-malleable code, though with a larger error.

Definition 1 can be readily extended to encompass arbitrary classes of tampering adversaries. However, for the sake of readability, we do not provide the generalization here. In the context of split-state tampering, we present the following two results.

Our first result gives an explicit average-case 2-split-state NMC for quantum messages with rate arbitrarily close to 1/11.

Theorem 1 (Average-Case 2-Split-State NMC for Quantum Messages with Constant Rate). *For any fixed constant* $\delta > 0$ *there exist an integer* $n_0 > 0$ *and* $c \in (0,1)$ *such that the following holds: There exists a family of coding schemes* $(\mathcal{C}_n)_{n\in\mathbb{N}}$ *where each* $\mathcal{C}_n$ *has codeword length* n *and message length* $\lfloor \left(\frac{1}{11} - \delta\right) n \rfloor$ *such that* C_n *is average-case* ε*-non-malleable for quantum messages with error* $\varepsilon = 2^{-n^c}$ *for all integers* $n \geq n_0$*. Furthermore, there exist encoding and decoding procedures for the family* $(\mathcal{C}_n)_{n\in\mathbb{N}}$ *running in time* $\mathrm{poly}(n)$*.*

Our second result, which builds on Theorem 1, gives an explicit construction of a worst-case 2-split-state NMC for quantum messages.

Theorem 2 (Worst-case 2-split-state NMC for quantum messages). *There exist constants* $c \in (0,1)$ *and* $n_0 \in \mathbb{N}$ *such that the following holds: There exists a family of coding schemes* $(\mathcal{C}_n)_{n\in\mathbb{N}}$ *where each* $\mathcal{C}_n$ *has codeword length* n *and message length* $\lfloor n^c \rfloor$ *such that* $\mathcal{C}_n$ *is* ε*-non-malleable for quantum messages with error* $\varepsilon = 2^{-n^c}$ *for all integers* $n \geq n_0$*. Furthermore, there exist encoding and decoding procedures for the family* $(\mathcal{C}_n)_{n\in\mathbb{N}}$ *running in time* $\mathrm{poly}(n)$*.*

In fact, we show something stronger: The explicit code from Theorem 2 is actually a 2-out-of-2 non-malleable secret sharing scheme for quantum messages with share size n, any message of length at most $n^{\Omega(1)}$, and error $\varepsilon = 2^{-n^{\Omega(1)}}$. We refer the reader to the full version for more details on non-malleable secret sharing [19].

Split-State Non-malleable Secret Sharing Schemes for Quantum Messages. The definition of threshold non-malleable secret sharing schemes for quantum messages is a natural and simple extension of our definition of 2-split-state non-malleable codes above, and it is analogous to the definition in the classical setting [42].

As our main result in this direction, which builds on Theorem 2, we construct efficient split-state NMSS schemes for quantum messages realizing more general threshold access structures with low privacy and non-malleability errors.

Theorem 3 (Split-State Threshold NMSS Schemes for Quantum Messages). *There exist constants $c, C > 0$ and an integer $n_0 \in \mathbb{N}$ such that the following holds for any number of parties p and threshold $t \geq 3$ such that $t \leq p \leq 2t - 1$ and for any $n \geq n_0$: There exists a family of t-out-of-p $(\varepsilon_{\mathsf{priv}} = \varepsilon, \varepsilon_{\mathsf{nm}} = \varepsilon)$-non-malleable secret sharing schemes for quantum messages with shares of size at most $(pn)^C$, message length $\lfloor n^c \rfloor$, and error $\varepsilon = 2^{-n^c}$. Furthermore, the sharing and reconstruction procedures of this scheme can be computed in time polynomial in p and n.*

Combined with our previously discussed 2-out-of-2 non-malleable secret sharing scheme for quantum messages, Theorem 3 covers all remaining threshold access structures for which secret sharing is possible in the quantum setting (i.e., those which do not violate no-cloning) except 2-out-of-3. We leave constructing a 2-out-of-3 non-malleable secret sharing scheme for quantum messages as a very interesting open problem. Finally, an analogous realization of the approach behind Theorem 3 using the same techniques allows us to obtain *classical* threshold non-malleable secret sharing schemes secure against quantum adversaries with shared entanglement.

See the full version for more details about our results on non-malleable secret sharing, including formal statements and proofs. In this abridged conference version, we provide only some informal discussion in the technical overview below.

1.2 Other Related Work

In this section, we discuss relevant prior work on classical NMCs and on quantum non-malleability beyond what we covered above.

Classical NMCs. We discuss prior work on NMCs for classical messages and secure against classical tampering adversaries. The first work on NMCs by Dziembowski, Pietrzak, and Wichs [37] showed that, surprisingly, there exists a (possibly inefficient) NMC against any family of at most $2^{2^{\alpha n}}$ tampering functions, where $\alpha < 1$ is an arbitrary constant, and n is the message length. The best possible rate of (possibly inefficient) NMCs was studied by Cheraghchi and Guruswami [30], who showed that split-state NMCs can have a rate of at most $1/2$. To complement the above, [37] constructed efficient NMCs in the *bit-wise tampering model*, a strictly weaker model than split-state tampering, where each bit of the codeword is tampered independently. As discussed before, this spurred a deep line of work which recently culminated in explicit constructions of constant-rate NMCs in the split-state model [6,7].

Several works have also studied NMCs against computationally-bounded adversaries from various hardness assumptions and setups, such as a common

reference string. Naturally, in such restricted settings, it is possible to achieve a better rate while allowing the adversary to perform many adaptive tamperings in a row. Computationally-restricted tampering models include polynomial-time algorithms [40], split-state polynomial-time adversaries [1], bounded-depth circuits [12,13,24], low-degree polynomials [11], decision trees [13,14], and streaming space-bounded algorithms [13,39]. Some other works have studied constructions of short NMCs based on conjectured properties of practical block ciphers [22,41] or extractable hash functions [48].

Classical NMSS Schemes. The notion of non-malleable secret sharing first appeared implicitly in the work of Aggarwal, Dziembowski, Kazana, and Obremski [5], where it was shown that every classical 2-split-state NMC is also a 2-out-of-2 secret sharing scheme with statistical privacy. NMSS schemes for classical messages and secure against classical adversaries were then studied explicitly and in much greater generality by Goyal and Kumar [42,43], leading to a long line of research on classical split-state non-malleable secret sharing, as mentioned above. Classical NMSS schemes have also been studied in tampering models beyond split-state tampering. For example, the original works of Goyal and Kumar [42,43] also consider a "joint" tampering model where, under certain restrictions, tampering adversaries may tamper with *subsets* of multiple shares, instead of only a single share. Stronger joint tampering models have been considered in the computational [21] and information-theoretic settings [45]. In an orthogonal direction, other notions of non-malleability where the adversary learns the reconstruction of tampered secrets with respect to multiple authorized subsets of parties have also been studied [3,21].

Other Notions of Quantum Non-malleability. Other notions of non-malleability for quantum messages have been studied in the context of *keyed* coding schemes by Ambainis, Bouda, and Winter [9] and Alagic and Majenz [8]. In the keyed setting, quantum authentication schemes [25] also provide non-malleability, since they allow one to detect tampering on the encoded quantum data.

We can view the setting studied in [8,9] for keyed coding schemes in the split-state model as follows: Let σ_M be the quantum message, and R be the key shared between the encoder and decoder. Let σ_M be encoded to $\rho_Z = \mathrm{Enc}_R(\sigma_M)$. Then, the adversary tampers $\rho_Z \to \tau_Z$, and the decoder outputs $\eta_M = \mathrm{Dec}_R(\tau_Z)$. Since the key R is available at both the encoder and decoder, we can view R as being the second part of the codeword, with the register Z being the first part of the codeword. Observe that in this model, the adversary is only allowed to tamper with the first part of the codeword, while our split-state tampering model allows the adversary to simultaneously but independently tamper with both parts of the codeword. Alagic and Majenz [8] used unitary 2-designs in their protocol for encoding and decoding. However, we note that their security definition involving mutual information appears a priori different from our Definition 1.

The question of whether one can even build split-state non-malleable coding schemes for quantum messages when there is no shared key and when the split-state adversary can tamper with both registers (with or without shared entanglement) remained open. In this work, we resolve this question in the affirmative.

Table 1 summarizes the main properties of known constructions of split-state NMCs and related constructions of keyed quantum schemes.

Table 1. Comparison between the best known explicit constructions of 2-split-state NMCs. Here, n denotes the codeword length.

Work by	Rate	Messages	Adversary	Shared key
[28]	$\frac{1}{\mathrm{poly}(n)}$	classical	classical	No
[50]	$\Omega\left(\frac{1}{\log n}\right)$	classical	classical	No
[51]	$\Omega\left(\frac{\log\log n}{\log n}\right)$	classical	classical	No
[7]	$\Omega(1)$	classical	classical	No
[52]	$\Omega(1)$	classical	classical	No
[6]	$1/3$	classical	classical	No
[2]	$\frac{1}{\mathrm{poly}(n)}$	classical	**quantum**	No
[16]	$\approx 1/5$	(average-case) classical	**quantum**	No
[8]	$\Omega(1)$	**quantum**	**quantum**	Yes
This work	$\approx 1/11$	**(average-case) quantum**	**quantum**	No
This work	$\frac{1}{\mathrm{poly}(n)}$	**quantum**	**quantum**	No

Concurrent Work. An earlier version of this work containing our results on NMCs for quantum messages was submitted to QCRYPT 2023 in April 2023 and presented there as a contributed talk. In work concurrent to and independent of that version, Bergamaschi [17] introduces, among other things, a natural quantum analogue of the *bitwise tampering* model [37], where each qubit of the encoding is tampered independently, and constructs high-rate codes in this setting satisfying a restricted form of keyless authentication called tamper detection, which is somewhat stronger than non-malleability. These contributions are incomparable to ours. First, we consider tampering adversaries with access to shared entanglement, while Bergamaschi [17] only studies the weaker setting where the various tampering adversaries are unentangled. In fact, tamper-detection codes (where the receiver either recovers the original message or aborts if they detect the adversary) are impossible to construct against adversaries with access to shared entanglement. This is because such adversaries can replace the quantum ciphertext with a fixed valid codeword. Second, the split-state tampering adversaries we study are much more powerful than bitwise tampering

adversaries. At a high level, both Bergamaschi's and our approaches rely on some combination of a non-malleable encoding of a classical key with an appropriate quantum encryption scheme. However, the two works differ significantly in how this approach is concretely realized and in the techniques used to establish the desired security properties.

As already discussed above, in another concurrent work Batra, Boddu, and Jain [16] constructed a classical 2-split-state non-malleable randomness encoder secure against quantum adversaries having shared entanglement with rate close to 1/2. They use it to build constant-rate quantum secure 3-split-state NMCs for quantum messages. Constructing 3-split-state NMCs is significantly easier than constructing 2-split-state NMCs (the hardest setting in the split-state tampering model) and the more general notion of split-state NMSS schemes for quantum messages, which are the focus of our work.

1.3 Technical Overview

Split-State NMCs for Quantum Messages. We follow a high-level strategy similar to that of Aggarwal, Agrawal, Gupta, Maji, Pandey, and Prabhakaran [1], who used authenticated encryption to transform any (augmented) split-state non-malleable code into a high-rate split-state non-malleable code resilient to computationally-bounded tampering in the classical setting. The main challenge we have to deal with, and which is not considered in [1], is that our tampering adversaries have quantum capabilities and are allowed to a priori share arbitrary *entangled quantum states*.

As a warm-up to our main construction, let us consider a simpler and natural approach that yields a 3-split-state non-malleable code for quantum messages. First, we observe that if we have a shared classical secret key R between the encoder and decoder, then we can detect whether *arbitrary* tampering has occurred by encoding the quantum message σ_M with a quantum authentication scheme $(\mathrm{Auth}_R, \mathrm{Ver}_R)$ [15,25]. Therefore, a reasonable approach would be to first encode the message using this quantum authentication scheme, then encode the key R using an existing split-state non-malleable code for classical messages, and lastly output the split-state encoding of R as part of the final codeword.

Fortunately, since R is a classical string, we can use the split-state non-malleable code $(\mathrm{cEnc}, \mathrm{cDec})$ of Aggarwal, Boddu, and Jain [2], which protects against quantum adversaries! Less fortunately, $\mathrm{cEnc}(R)$ yields a codeword with two parts, call them R_1 and R_2, that cannot be stored together.

To elaborate on this approach, let's define a quantum authentication scheme as $(\mathrm{Auth}_R, \mathrm{Ver}_R)$, where Auth_R is a quantum encoding procedure that takes a quantum state σ_M and a classical secret key R and outputs an authenticated state $\psi = \mathrm{Auth}_R(\sigma_M)$. The verification procedure Ver_R takes an authenticated state ψ' and the secret key R as input, and outputs either 1 (accept) or 0 (reject). The overall encoding procedure $\mathrm{Enc}(\sigma_M)$ for our 3-split-state non-malleable code is as follows:

1. Sample a classical secret key R uniformly at random from an appropriate keyspace.

2. Compute the authenticated state $\psi = \mathrm{Auth}_R(\sigma_M)$ using the quantum authentication scheme.
3. Compute the classical split-state encoding $(R_1, R_2) = \mathrm{cEnc}(R)$.
4. Output the 3-state codeword (ψ, R_1, R_2).

To decode a possibly tampered codeword (ψ', R_1', R_2'), the decoding procedure Dec works as follows:

1. Use the classical decoder $\mathrm{cDec}(R_1', R_2')$ to obtain a candidate key R'.
2. Run the verification procedure $\mathrm{Ver}_{R'}(\psi')$ to check the authenticity of the quantum state ψ' using the candidate key R'.
3. If the verification procedure outputs 1 (accept), then output $\mathrm{Dec}(\psi', R_1', R_2')$. Otherwise, output a special symbol to indicate tampering.

This approach leverages the quantum authentication scheme to detect any tampering of the quantum message, and the classical split-state non-malleable code to protect the classical key R against tampering. Intuitively, if we consider quantum adversaries *without* shared entanglement, the non-malleability of the 3-state coding scheme can be understood as follows: If the adversary attempts to tampering with R_1 and R_2, then the properties of the classical split-state non-malleable code $(\mathrm{cEnc}, \mathrm{cDec})$ guarantee that either the candidate key R' remains unchanged $(R' = R)$ or that it becomes independent of R. In the former case, when the decoder Dec calls $\mathrm{Ver}_R(\psi')$, the quantum authentication scheme will, with high probability, detect whether $\psi' \neq \psi$, indicating tampering, and Dec will abort the decoding process. In the latter case, the decoder calls $\mathrm{Ver}_{R'}(\psi')$ with R' being independent of R and, consequently, independent of ψ' as well. Choosing a quantum authentication scheme, such as Clifford-based authentication, it is possible to show that $\mathrm{Ver}_{R'}(\psi')$ will output a state independent of (and thus unrelated to) the message σ with high probability.

Overall, this approach combines the strength of quantum authentication to detect tampering with the quantum message and the non-malleability property of the classical split-state code to protect the classical key R against tampering. By leveraging these properties together, we obtain a 3-split-state non-malleable code for quantum messages. Although the coding scheme above already gives some non-malleability guarantees for quantum messages, it features some major shortcomings. First, it requires dividing the codeword into three parts which must be stored separately. Ideally, we would like to construct efficient coding schemes that only need to be divided into *two* parts. Second, our argument above only works against quantum adversaries *without* shared entanglement. However, as we move to the more powerful setting with adversaries having shared entanglement, we need to develop more sophisticated techniques to ensure non-malleability.

Split-State Non-malleable Codes for Average-Case Quantum Messages. Perhaps the most natural approach to building a split-state non-malleable code secure against quantum adversaries with shared entanglement would be to take our

3-state code above and merge two of the parts. More precisely, we could attempt to analyze the code which outputs

$$(\psi, R_1) \quad \text{and} \quad R_2$$

as its two parts, where we recall that ψ is the authenticated state via the secret key R and $(R_1, R_2) = \text{cEnc}(R)$ is the classical split-state non-malleable encoding of R.

This matches the approach taken in [1]. However, as already mentioned above, they did not have to handle quantum adversaries with shared entanglement. Although this is also essentially the approach we successfully undertake, establishing non-malleability of this modified coding scheme is significantly more involved than the intuitive argument laid out above, and also requires some small further changes to the scheme. Indeed, one of the first difficulties arises due to the use of trap flags[5] in the authentication schemes, as we will discuss subsequently. We note that, unlike in an authentication scheme, the decoder need not output $\perp$ in a non-malleable code, since error detection is not required. We therefore get rid of the trap flags used in the authentication scheme specified in the 3-split argument above, and this is one of the first insights that allows our analysis to work.

We first describe a sub-optimal version of our approach in the setting of *average-case non-malleability*, and then discuss how its rate can be optimized. Per Definition 1, this corresponds to the scenario where the message σ_M is assumed to be a maximally mixed quantum state with canonical purification $\sigma_{M\hat{M}}$. To construct our split-state non-malleable code for quantum messages, we use random Clifford unitaries $\{(C_r, C_r^\dagger)\}_{r \leftarrow R}$ with underlying classical randomness R, along with the classical split-state coding scheme $(\text{cEnc}, \text{cDec})$ designed in [2] as a quantum-secure non-malleable code for the classical string R.

Inspired by the 3-state approach above, we use $(\text{cEnc}, \text{cDec})$ to protect the key R in a non-malleable manner, and then use the random Clifford C_R to protect the quantum message σ_M. This yields the following encoding procedure $\text{Enc}(\sigma_M)$, where we use slightly different notation than the above to facilitate our analysis (see also Fig. 4 for a diagram of our encoding and decoding procedures):

1. Sample a classical secret key R uniformly at random (independent of $\sigma_{M\hat{M}}$) from an appropriate keyspace;
2. Compute the state $(\sigma_1)_Z = C_R(\sigma_M)C_R^\dagger$;
3. Compute the split-state encoding $\text{cEnc}(R) = (\sigma_1)_{XY}$;
4. Output the 2-part codeword $((\sigma_1)_{ZX}, (\sigma_1)_Y)$.

[5] Clifford-based quantum authentication schemes apply a random (secret) Clifford operator to the message plus several additional "trap registers" initialized to $|0\rangle$. Verifying whether tampering of the authenticated state occurred consists of checking whether the trap registers all return to the $|0\rangle$ state after applying the inverse Clifford operator. If this does not hold, then the verification procedure outputs the special symbol $\perp$, which we call the "trap flag".

The message $\sigma_M = U_M$, along with its purification register $\hat{M}$, is thus encoded into $(\sigma_1)_{\hat{M}ZXY}$.

As our first observation, note that the register Z, which holds the Clifford-protected message, may carry information about the classical key R, since $(\sigma_1)_Z = C_R(\sigma_M)C_R^\dagger$. Fortunately, since $\sigma_M = U_M$, where U_M is a maximally mixed state, we have

$$(\sigma_1)_{ZXY} = (\sigma_1)_Z \otimes (\sigma_1)_{XY}.$$

One may then expect that we can use the argument of [2] for classical messages to argue that the key R remains secure even if the adversary sees the register Z when tampering one of the parts of $\mathrm{cEnc}(R)$. Unfortunately, the argument of [2] does not go through in this scenario. Namely, for $(\mathrm{cEnc}, \mathrm{cDec})$ to protect the key R in a non-malleable manner after adversarial tampering we need that

$$(\sigma_1)_{\hat{M}ZXY} = (\sigma_1)_{\hat{M}Z} \otimes (\sigma_1)_{XY}.$$

However, it can be verified that the registers $\hat{M}Z$ are not independent of XY in state σ_1. To circumvent this issue, we use the transpose method (see Fact 5) for state $\sigma_{M\hat{M}}$, and note that the application of the random Clifford gate C_R and the adversarial operations commute in σ_1 (see Figs. 4 and 5). This allows us to delay the operation C_R on register $\hat{M}$ (see Fig. 6). Crucially, we could not have used the transpose method in the presence of flag registers as employed in quantum authentication.

Carefully combining the above with arguments from [2], we conclude that $(\mathrm{cEnc}, \mathrm{cDec})$ can protect the key R after adversarial tampering on state θ_1 (which corresponds to the state prior to applying the C_R operation on register $\hat{M}$ in Fig. 6), since

$$(\theta_1)_{\hat{M}ZXY} = (\theta_1)_{\hat{M}Z} \otimes (\theta_1)_{XY}.$$

Let θ_2 be the state obtained after adversarial tampering on θ_1. By the properties of $(\mathrm{cEnc}, \mathrm{cDec})$, we are essentially guaranteed that either $R' = R$ or that R' is independent of R in θ_2. However, this is not enough, and we observe that $(\mathrm{cEnc}, \mathrm{cDec})$ actually guarantees something even stronger! More precisely, in state θ_2 we have either:

- $R = R'$ and R is independent of registers $Z\hat{M}$;
- R is independent of $R'Z\hat{M}$.

In the case where R is independent of $R'Z\hat{M}$, applying the delayed operation C_R on register $\hat{M}$ decouples the registers $R'Z \otimes \hat{M}$, and so we are done. We are thus left to analyze the case where $R = R'$ and R is independent of registers $Z\hat{M}$. Here, we make use of the 2-design properties of the Clifford scheme $(C_R, C_R^\dagger)$. Roughly speaking, first suppose that the adversary applied $\mathbb{I}_Z$ on register Z. Then, since the message was maximally mixed, we conclude that we get an EPR state as the outcome of the tampering experiment (after the decoding procedure is applied). Now, suppose that the adversary applied some $P \neq \mathbb{I}_Z$ on register

Z. We make use of Clifford randomization and the twirl property to handle this scenario. To elaborate, first note that the state

$$(\mathbb{I} \otimes P)(\theta_1)_{\hat{M}Z}(\mathbb{I} \otimes P^\dagger)$$

is in a subspace orthogonal to an EPR state in $\hat{M}Z$. Consider the state after applying the delayed operation C_R on register $\hat{M}$ and $C_R^\dagger$ on register Z. Here, Clifford randomization and the twirl property ensure that any state orthogonal to an EPR state in registers $\hat{M}Z$ is exactly maximally mixed in a subspace (of dimension $4^{|\hat{M}|} - 1$) orthogonal to this EPR state in registers $\hat{M}Z$. As a result, the outcome of the tampering experiment after the decoding procedure is applied is close in trace distance to $U_{\hat{M}} \otimes U_M$. Our proof overall crucially uses interesting properties of the Pauli and Clifford unitaries including Pauli twirl, Clifford twirl, and Clifford randomization.

Improving the Rate from Sub-constant to Constant in the Average-Case Setting. An important quantity associated with a (non-malleable code) is its *rate*—the ratio between the size of a message and the size of its corresponding encoding. Looking at the proposed approach we discussed above, we conclude that its rate is sub-constant, i.e., the size of the encoding is a superlinear function of the message size. The main reasons behind this are as follows: First, the classical key R that we use to sample the Clifford operator C_R is much longer than the message σ_M (In fact $|R| = \mathcal{O}(|M|^2)$). Second, the rate of the underlying non-malleable code for classical messages $(\mathsf{cEnc}, \mathsf{cDec})$ from [2] is already sub-constant.

We now explain how our approach above can be modified to yield a constant-rate average-case non-malleable code for quantum messages. First, instead of using the whole Clifford group, we can use a shorter random key R to sample a Clifford operator from a smaller subgroup with special properties that suffice for our needs. A result of Cleve, Leung, Liu, and Wang [33] guarantees that this can be done efficiently with a classical key R of length at most $5|M|$. Second, observe that we only care about obtaining a split-state non-malleable encoding of a *uniformly random* classical key R. This means that we can replace the classical split-state non-malleable code $(\mathsf{cEnc}, \mathsf{cDec})$ from [2] by a *Non-Malleable Randomness Encoder* (NMRE), an object originally introduced by Kanukurthi, Obbattu, and Sekar [47] and whose known constructions enjoy much better rates than non-malleable codes. Batra, Boddu, and Jain [16] recently constructed a classical NMRE secure against quantum adversaries having shared entanglement with a rate close to 1/2.[6] In contrast, as mentioned above, the NMC for classical messages from [2] only has a sub-constant rate.

[6] For the experienced reader, Batra, Boddu, and Jain [16] construct an explicit quantum-secure 2-source non-malleable extractor nmExt with a large output length. We can then sample classical bitstrings X and Y uniformly at random with appropriate lengths and set the classical key R to be $R = \mathsf{nmExt}(X, Y)$; this is the quantum-secure classical NMRE that we use in our optimized coding scheme.

Using these two results in our previously described approach allows us to improve the rate of our average-case NMC for quantum messages from sub-constant to close to 1/11.

From Average-Case to Worst-Case NMCs for Quantum Messages. In the discussion above we assumed that the quantum message was maximally mixed. However, we would like to show that our code is a worst-case NMC, i.e., it is non-malleable for *any* fixed quantum message.

Consider an arbitrary quantum message ρ_M with canonical purification $\rho_{M\hat{M}}$. Recall that $\sigma_{\hat{M}M}$ is the canonical purification of $\sigma_M = U_M$. Using a quantum rejection sampling argument, we obtain a measurement acting on register $\hat{M}$ of σ such that the state conditioned on "success" is exactly ρ. Moreover, this measurement on the register $\hat{M}$ commutes with Enc, Dec, and the adversarial operations, and it succeeds with probability $2^{-|M|}$. If the error of the underlying average-case NMC is ε, then this measurement allows us to argue non-malleability of the same NMC for any message ρ_M (i.e., worst-case non-malleability) with larger error $\varepsilon' = 2^{|M|} \cdot \varepsilon$, and we can easily handle this blow-up in the error at the expense of dropping the rate of the code from constant to sub-constant. Intuitively, this happens because of the following: Suppose that there is a fixed message ρ for which Equation (1) (with ρ in place of σ) holds only with error larger than ε'. Therefore, when faced with a maximally mixed message $\sigma_{M\hat{M}}$, we can apply the measurement above to $\hat{M}$ and, if the measurement succeeds and the resulting state is ρ, distinguish with advantage larger than ε'. Since the measurement succeeds with probability at least $2^{-|M|}$, the overall distinguishing advantage is larger than $2^{-|M|} \cdot \varepsilon' = \varepsilon$, which contradicts the average-case ε-non-malleability of the NMC.

Threshold Split-State NMSS Schemes for Quantum Messages. At a high level, in order to construct our threshold NMSS schemes for quantum messages, we combine our split-state NMC for quantum messages above with the approach of Goyal and Kumar [42] used to construct NMSS schemes in the classical setting. However, as we will discuss, following this approach in the quantum setting poses various challenges.

Roughly speaking, the approach of Goyal and Kumar [42] for building NMSS schemes proceeds as follows for p parties and a threshold $3 \leq t \leq p$. On input a message m, first encode it with the split-state NMC to generate two states, L and R. Now, apply a standard t-out-of-p secret sharing scheme to L (say, Shamir's secret sharing), yielding shares $L_1, \ldots, L_p$. Furthermore, apply a 2-out-of-p *leakage-resilient* secret sharing scheme to R, yielding shares $R_1, \ldots, R_p$. Intuitively, a secret sharing scheme is leakage-resilient if the input remains private even when the adversary learns an unauthorized subset of shares *plus* bounded side information from every other share. Finally, set the resulting i-th share S_i as $S_i = (L_i, R_i)$ for each $i \in [p]$.

To argue the non-malleability of this construction, Goyal and Kumar showed how to transform any tampering attack on the resulting secret sharing scheme into an essentially-as-good tampering attack on the underlying 2-split-state NMC. The main challenge in designing such a reduction is that the two tampering functions for the underlying NMC must act independently – we must tamper L without knowledge of R, and vice versa. On the other hand, the i-th tampering function for the secret sharing scheme tampers (L_i, R_i) into (L'_i, R'_i), and so potentially has access to some information from both L and R. For simplicity, let L' and R' denote the tampered secrets reconstructed from $(L'_1, \ldots, L'_t)$ and (R'_1, R'_2), respectively.

Showing that we can obtain R' from R without knowing L is easy. This follows because R' is fully determined by R'_1 and R'_2, which in turn only depend on L_1 and L_2. Since the L_i's are a t-out-of-p Shamir secret sharing of L with threshold $t \geq 3$, the two shares L_1 and L_2 are independent of L. However, arguing that we can (up to small error) obtain L' from L without knowledge of R is a lot trickier. The previous argument clearly does not immediately work since the t shares $L'_1, \ldots, L'_t$ depend on $R_1, \ldots, R_t$, respectively, which determine R. This is where the leakage-resilience property kicks in – if we see the L'_is as bounded leakage on the R_is, then leakage-resilience guarantees that R is (close to) independent of the leakages $L'_1, \ldots, L'_t$, and so is independent of L'.

Realizing This Approach in the Quantum Setting. We follow the same high-level approach for a quantum message σ_M. To that end, we first replace the classical split-state NMC by our NMC for quantum messages discussed previously. We use additional key properties of our NMC: (1) The resulting left state L is quantum, but the right state R is *classical*, and (2) we prove that it is actually a 2-out-of-2 split-state NMSS scheme, and so, in particular, learning only one of L and R reveals nothing about σ_M. Since L is quantum, we now apply a standard t-out-of-p (for $p < 2t$) secret sharing scheme for quantum messages, such as "quantum" Shamir secret sharing [32], to get quantum shares $(L_1, \ldots, L_p)$. And, since R is classical, we secret-share it using a 2-out-of-p scheme satisfying a special leakage-resilience property that we will determine later.

Establishing correctness and privacy of the resulting t-out-of-p secret sharing scheme is not difficult using property (2) of our NMC above. It remains to prove non-malleability. Arguing that R' can be obtained from R without knowledge of L still follows easily from the fact that R is shared using a 2-out-of-p scheme, while L is shared using a t-out-of-p scheme with $t \geq 3$. Here, it is crucial that R is classical. Otherwise, a 2-out-of-p scheme would not exist when $p \geq 4$, and this means that we would not be able to, say, construct $(p/2+1)$-out-of-p split-state NMSS schemes for quantum messages for *any* even p using this approach.

We also want to argue that L' can be obtained from L without knowledge of R. As before, we would like to see $L'_1, \ldots, L'_p$ as leakages on the secret shares $R_1, \ldots, R_p$, and then exploit the leakage-resilience of the scheme used to share R to conclude that R is independent of $L'_1, \ldots, L'_p$, and hence of L'. However, realizing this in our quantum setting requires stronger leakage-resilience prop-

erties: First, the tamperings $L'_1, \ldots, L'_p$ are now quantum states. Second, the tampering functions in our setting *share arbitrary entangled states*. This means that the i-th tampering function now sees (L_i, R_i, W_i), where $W_1, \ldots, W_p$ are quantum registers holding an arbitrary state. To overcome these barriers, we introduce *augmented* leakage-resilient secret sharing schemes. This corresponds to a setting where there are p local adversaries $\mathcal{A}_1, \ldots, \mathcal{A}_p$ sharing an arbitrary entangled state spread across registers $W_1, \ldots, W_p$, respectively, and each one having access to a share $R_1, \ldots, R_p$. We require that R remains hidden even if $\mathcal{A}_i$ knows the share R_i, local bounded quantum leakages $(\mathsf{Leak}_j(R_j, W_j))_{j \in [p] \setminus \{i\}}$ from every other share, *and also the entangled state* W_i (hence the "augmented" adjective). We prove that the 2-out-of-2 secret sharing scheme whereby R is shared into X and Y sampled uniformly such that $\langle X, Y \rangle = R$ is augmented leakage-resilient with good parameters. This relies on the randomness extraction properties of the inner product function and the formalism of *qpa-states*[7] from [20]. We can then extend this scheme to a 2-out-of-p augmented leakage-resilient scheme for classical messages in a standard manner. Although notions of leakage-resilience against quantum adversaries with shared entanglement have been studied in other recent work [26], these do not cover augmented leakage-resilience.

Besides the above, proving non-malleability requires dealing with additional subtleties specific to the quantum setting. The original non-malleability argument in [42] proceeds by fixing the values of certain components (e.g., shares, leakages). However, we cannot fully emulate this approach in the quantum setting: The left state L is quantum, and so are the bounded leakages that show up in the analysis. Therefore, we cannot fix them. Moreover, again because L is quantum, it is modified after the tampering functions are applied, but we still need to access the "original" L in the analysis. For this, with some work, we can use the message's canonical purification register $\hat{M}$ to generate a new register $\hat{L}$ which can be thought of as a coherent copy of the original left state L.

1.4 Open Problems

We list here some interesting directions for future research:

- Can we design (worst-case) split-state NMCs for quantum messages with a constant rate? This is open even for classical messages against quantum adversaries with shared entanglement. More ambitiously, can we construct (worst-case) split-state NMSS schemes for quantum messages with a constant rate?
- Can we construct 2-out-of-3 split-state non-malleable secret sharing schemes for quantum messages?
- Can we design NMSS schemes for quantum messages that are secure against joint tampering of shares?
- What can we achieve if we consider computationally-bounded adversaries instead?

[7] "qpa-state" stands for quantum purified adversary state.

– Besides the direct application to tamper-proof distributed storage, can we find additional applications of non-malleable codes and secret sharing schemes for quantum messages, perhaps analogous to those found in the classical setting?

2 Preliminaries

This section collects basic notation and conventions alongside useful facts and lemmas that we use in the proofs of our main results. In this work, facts denote results already known from prior work, and lemmas denote auxiliary results that we prove here.

2.1 Basic General Notation

All the logarithms are evaluated to the base 2. We denote sets by uppercase calligraphic letters such as $\mathcal{X}$ and use uppercase roman letters such as X and Y for both random variables and quantum registers. The distinction will be clear from context. The set $\{1, \ldots, n\}$ may be written as $[n]$, and we may also more generally write $[t, n]$ for the set $\{t, t+1, \ldots, n\}$. We denote the uniform distribution over $\{0,1\}^d$ by U_d. For a *random variable* $X \in \mathcal{X}$, we use X to denote both the random variable and its distribution, whenever it is clear from context. We use $x \leftarrow X$ to denote that x is drawn according to X, and, for a finite set $\mathcal{X}$, we use $x \leftarrow \mathcal{X}$ to denote that x is drawn uniformly at random from $\mathcal{X}$. For two random variables X, Y we use $X \otimes Y$ to denote their product distribution. We call random variables X and Y *copies* of each other if and only if $\Pr[X = Y] = 1$.

2.2 Quantum Information Theory

In this section we cover some important basic prerequisites from quantum information theory alongside some useful lemmas and facts.

Conventions and Notation. Consider a finite-dimensional Hilbert space $\mathcal{H}$ endowed with an inner-product $\langle \cdot, \cdot \rangle$ (we only consider finite-dimensional Hilbert-spaces). A quantum state (or a density matrix or a state) is a positive semi-definite operator on $\mathcal{H}$ with trace value equal to 1. It is called *pure* if and only if its rank is 1. Let $|\psi\rangle$ be a unit vector on $\mathcal{H}$, that is $\langle \psi, \psi \rangle = 1$. With some abuse of notation, we use ψ to represent the state and also the density matrix $|\psi\rangle\langle\psi|$ associated with $|\psi\rangle$. Given a quantum state ρ on $\mathcal{H}$, the *support of* ρ, denoted by $\mathrm{supp}(\rho)$, is the subspace of $\mathcal{H}$ spanned by all eigenvectors of ρ with non-zero eigenvalues.

A *quantum register* A is associated with some Hilbert space $\mathcal{H}_A$. Define $|A| := \log(\dim(\mathcal{H}_A))$. For a sequence of registers $A_1, \ldots, A_n$ and a set $T \subseteq [n]$, we define the projection according to T as $A_T = (A_i)_{i \in T}$. Let $\mathcal{L}(\mathcal{H}_A)$ represent the set of all linear operators on the Hilbert space $\mathcal{H}_A$. For an operator $O \in \mathcal{L}(\mathcal{H}_A)$, we use

O^T to represent the transpose of O. For operators $O, O' \in \mathcal{L}(\mathcal{H}_A)$, the notation $O \leq O'$ represents the Löwner order, that is, $O' - O$ is a positive semi-definite operator. We denote by $\mathcal{D}(\mathcal{H}_A)$ the set of all quantum states on the Hilbert space $\mathcal{H}_A$. The state ρ with subscript A indicates that $\rho_A \in \mathcal{D}(\mathcal{H}_A)$. If two registers A, B are associated with the same Hilbert space, we shall represent the relation by $A \equiv B$. For two states ρ and σ, we write $\rho \equiv \sigma$ if they are identical as states (potentially in different registers). Composition of two registers A and B, denoted AB, is associated with the Hilbert space $\mathcal{H}_A \otimes \mathcal{H}_B$. For two quantum states $\rho \in \mathcal{D}(\mathcal{H}_A)$ and $\sigma \in \mathcal{D}(\mathcal{H}_B)$, $\rho \otimes \sigma \in \mathcal{D}(\mathcal{H}_{AB})$ represents the tensor product (*Kronecker* product) of ρ and σ. The identity operator on $\mathcal{H}_A$ is denoted $\mathbb{I}_A$. Let U_A denote the maximally mixed state in $\mathcal{H}_A$. Let $\rho_{AB} \in \mathcal{D}(\mathcal{H}_{AB})$. Define

$$\rho_B \stackrel{\text{def}}{=} \text{Tr}_A \rho_{AB} \stackrel{\text{def}}{=} \sum_i (\langle i| \otimes \mathbb{I}_B) \rho_{AB} (|i\rangle \otimes \mathbb{I}_B),$$

where $\{|i\rangle\}_i$ is an orthonormal basis for the Hilbert space $\mathcal{H}_A$. The state $\rho_B \in \mathcal{D}(\mathcal{H}_B)$ is referred to as the marginal state of ρ_{AB} on the register B. Unless otherwise stated, a missing register from subscript in a state represents partial trace over that register. Given $\rho_A \in \mathcal{D}(\mathcal{H}_A)$, a *purification* of ρ_A is a pure state $\rho_{AB} \in \mathcal{D}(\mathcal{H}_{AB})$ such that $\text{Tr}_B \rho_{AB} = \rho_A$. Purification of a quantum state is not unique. Suppose $A \equiv B$. Given $\{|i\rangle_A\}$ and $\{|i\rangle_B\}$ as orthonormal bases over $\mathcal{H}_A$ and $\mathcal{H}_B$ respectively, the *canonical purification* of a quantum state ρ_A is a pure state $\rho_{AB} \stackrel{\text{def}}{=} (\rho_A^{\frac{1}{2}} \otimes \mathbb{I}_B) (\sum_i |i\rangle_A |i\rangle_B)$.

A quantum map $\mathcal{E} : \mathcal{L}(\mathcal{H}_A) \to \mathcal{L}(\mathcal{H}_B)$ is a completely positive and trace preserving (CPTP) linear map. A CPTP map $\mathcal{E}$ is described by the Kraus operators $\{M_i : \mathcal{H}_A \to \mathcal{H}_B\}_i$ such that $\mathcal{E}(\rho) = \sum_i M_i \rho M_i^\dagger$ and $\sum_i M_i^\dagger M_i = \mathbb{I}_A$. A *Hermitian* operator $H : \mathcal{H}_A \to \mathcal{H}_A$ is such that $H = H^\dagger$. A projector $\Pi \in \mathcal{L}(\mathcal{H}_A)$ is a Hermitian operator such that $\Pi^2 = \Pi$. A *unitary* operator $V_A : \mathcal{H}_A \to \mathcal{H}_A$ is such that $V_A^\dagger V_A = V_A V_A^\dagger = \mathbb{I}_A$. The set of all unitary operators on $\mathcal{H}_A$ is denoted by $\mathcal{U}(\mathcal{H}_A)$. An *isometry* $V : \mathcal{H}_A \to \mathcal{H}_B$ is such that $V^\dagger V = \mathbb{I}_A$. A *POVM* element is an operator $0 \leq M \leq \mathbb{I}$. We use the shorthand $\bar{M} \stackrel{\text{def}}{=} \mathbb{I} - M$, where $\mathbb{I}$ is clear from context. We use shorthand M to represent $M \otimes \mathbb{I}$, where $\mathbb{I}$ is clear from context.

Registers, Quantum Maps, and Isometries. This section collects definitions of certain registers and operations on them.

Definition 2 (Classical Register in a Pure State). *Let $\mathcal{X}$ be a set. A* classical-quantum *(c-q) state ρ_{XE} is of the form $\rho_{XE} = \sum_{x \in \mathcal{X}} p(x) |x\rangle\langle x| \otimes \rho_E^x$, where ρ_E^x are states.*

Let ρ_{XEA} be a pure state. We call X a classical register *in ρ_{XEA} if ρ_{XE} (or ρ_{XA}) is a c-q state. Whenever it is clear from context, we identify the random variable X with the register X via $\Pr[X = x] = p(x)$.*

Definition 3 (Copy of a Classical Register). *Let $\rho_{X\hat{X}E}$ be a pure state with X being a classical register in $\rho_{X\hat{X}E}$ taking values in $\mathcal{X}$. Similarly, let $\hat{X}$ be a*

classical register in $\rho_{X\hat{X}E}$ *taking values in* $\mathcal{X}$. *Let* $\Pi_{\mathsf{Eq}} = \sum_{x\in\mathcal{X}} |x\rangle\langle x| \otimes |x\rangle\langle x|$ *be the* equality *projector acting on the registers* $X\hat{X}$. *We call* X *and* $\hat{X}$ *copies of each other (in the computational basis) if* $\mathrm{Tr}\left(\Pi_{\mathsf{Eq}}\rho_{X\hat{X}}\right) = 1$.

Definition 4 (Conditioning). *Let* $\rho_{XE} = \sum_{x\in\{0,1\}^n} p(x)|x\rangle\langle x|\otimes\rho_E^x$ *be a c-q state. For an event* $\mathcal{S} \subseteq \{0,1\}^n$, *define*

$$\Pr[\mathcal{S}]_\rho \stackrel{\text{def}}{=} \sum_{x\in\mathcal{S}} p(x) \quad \textit{and} \quad (\rho|X \in \mathcal{S}) \stackrel{\text{def}}{=} \frac{1}{\Pr[\mathcal{S}]_\rho} \sum_{x\in\mathcal{S}} p(x)|x\rangle\langle x|\otimes\rho_E^x.$$

We sometimes shorthand $(\rho|X \in \mathcal{S})$ *as* $(\rho|\mathcal{S})$ *when the register* X *is clear from context.*

Let ρ_{AB} *be a state with* $|A| = n$. *We define* $(\rho|A \in \mathcal{S}) \stackrel{\text{def}}{=} (\sigma|\mathcal{S})$, *where* σ_{AB} *is the c-q state obtained by measuring the register* A *in* ρ_{AB} *in the computational basis. In the case where* $\mathcal{S} = \{s\}$ *is a singleton set, we shorthand* $(\rho|A = s) \stackrel{\text{def}}{=} \mathrm{Tr}_A(\rho|A = s)$.

Definition 5 (Safe Maps). *We call an isometry* $V : \mathcal{H}_X \otimes \mathcal{H}_A \to \mathcal{H}_X \otimes \mathcal{H}_B$, safe *on* X *if and only if there is a collection of isometries* $V_x : \mathcal{H}_A \to \mathcal{H}_B$ *such that for all states* $|\psi\rangle_{XA} = \sum_x \alpha_x|x\rangle_X|\psi^x\rangle_A$ *we have that*

$$V|\psi\rangle_{XA} = \sum_x \alpha_x|x\rangle_X V_x|\psi^x\rangle_A.$$

Definition 6 (Extension). *Let* $\rho_{XE} = \sum_{x\in\mathcal{X}} p(x)|x\rangle\langle x|\otimes\rho_E^x$ *be a c-q state. For a function* $Z : \mathcal{X} \to \mathcal{Z}$, *define the following extension of* ρ_{XE},

$$\rho_{ZXE} \stackrel{\text{def}}{=} \sum_{x\in\mathcal{X}} p(x)|Z(x)\rangle\langle Z(x)|\otimes|x\rangle\langle x|\otimes\rho_E^x.$$

For a pure state ρ_{XEA} (with X classical and $X \in \mathcal{X}$) and a function $Z : \mathcal{X} \to \mathcal{Z}$, define $\rho_{Z\hat{Z}XEA}$ to be a pure state extension of ρ_{XEA} generated via a safe isometry $V : \mathcal{H}_X \to \mathcal{H}_X \otimes \mathcal{H}_Z \otimes \mathcal{H}_{\hat{Z}}$ (Z classical with copy $\hat{Z}$). We use the notation $\mathcal{M}_A(\rho_{AB})$ to denote measurement in the computational basis on register A in state ρ_{AB}.

All isometries considered in this paper are safe on classical registers that they act on. Isometries applied by adversaries can be assumed without loss of generality as safe on classical registers, by the adversary first making a (safe) copy of classical registers and then proceeding as before. This does not reduce the power of the adversary.

Norms, Trace Distance, and Divergences. This section collects definitions of some important quantum information-theoretic quantities and related useful properties.

Definition 7 (Schatten p-Norm). *For* $p \geq 1$ *and a matrix* A, *the* Schatten p-norm *of* A, *denoted by* $\|A\|_p$, *is defined as* $\|A\|_p \stackrel{\text{def}}{=} (\mathrm{Tr}(A^\dagger A)^{\frac{p}{2}})^{\frac{1}{p}}$.

Definition 8 (Trace Distance). *The* trace distance *between two states ρ and σ is given by $\|\rho - \sigma\|_1$. We write $\rho \approx_\varepsilon \sigma$ if $\|\rho - \sigma\|_1 \leq \varepsilon$.*

In the classical setting the trace distance corresponds to the *statistical distance* (a.k.a. total variation distance). We write $X \approx_\varepsilon Y$ for two random variables X and Y if the statistical distance between their distributions is at most ε.

Definition 9 (Max-Divergence ([35], see also [46])). *Given states ρ and σ such that* $\mathrm{supp}(\rho) \subseteq \mathrm{supp}(\sigma)$*, the* max-divergence *between ρ and σ, denoted by $\mathrm{D}_{\max}(\rho\|\sigma)$, is defined as $\mathrm{D}_{\max}(\rho\|\sigma) = \min\{\lambda \in \mathbb{R} : \rho \leq 2^\lambda \sigma\}$.*

For the facts stated below without citation, we refer the reader to standard textbooks [54,58]. The following facts state some basic properties of trace distance.

Fact 1 (Data-Processing Inequality). *Let ρ, σ be states and $\mathcal{E}$ be a CPTP map. Then, $\|\mathcal{E}(\rho) - \mathcal{E}(\sigma)\|_1 \leq \|\rho - \sigma\|_1$. This inequality is an equality whenever $\mathcal{E}$ is a CPTP map corresponding to an isometry.*

Pauli and Clifford Operators. We proceed to define Pauli operators and the associated Pauli and Clifford groups.

Definition 10 (Pauli operators). *The single-qubit* Pauli operators *are given by*

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \quad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

An n-qubit Pauli operator is given by the n-fold tensor product of single-qubit Pauli operators. We denote the set of all $|A|$-qubit Pauli operators on $\mathcal{H}_A$ by $\mathcal{P}(\mathcal{H}_A)$, where $|\mathcal{P}(\mathcal{H}_A)| = 4^{|A|}$. Any linear operator $L \in \mathcal{L}(\mathcal{H}_A)$ can be written as a linear combination of $|A|$-qubit Pauli operators with complex coefficients as $L = \sum_{P \in \mathcal{P}(\mathcal{H}_A)} \alpha_P P$. This is called the Pauli decomposition *of a linear operator.*

Definition 11 (Pauli Group). *The single-qubit* Pauli group *is given by*

$$\{+P, -P,\ iP,\ -iP : P \in \{I, X, Y, Z\}\}.$$

The Pauli group on $|A|$-qubits is the group generated by the operators described above applied to each of $|A|$-qubits in the tensor product. We denote the $|A|$-qubit Pauli group on $\mathcal{H}_A$ by $\tilde{\mathcal{P}}(\mathcal{H}_A)$.

Definition 12 (Clifford Group). *The* Clifford group $\mathcal{C}(\mathcal{H}_A)$ *is defined as the group of unitaries that normalize the Pauli group $\tilde{\mathcal{P}}(\mathcal{H}_A)$, i.e.,*

$$\mathcal{C}(\mathcal{H}_A) = \{V \in \mathcal{U}(\mathcal{H}_A) : V\tilde{\mathcal{P}}(\mathcal{H}_A)V^\dagger = \tilde{\mathcal{P}}(\mathcal{H}_A)\}.$$

The Clifford unitaries *are the elements of the Clifford group.*

We will also need to work with subgroups of the Clifford group with certain special properties. The following fact describes these properties and guarantees the existence of such subgroups.

Fact 2 (Subgroup of the Clifford group [33]). *There exists a subgroup $\mathcal{SC}(\mathcal{H}_A)$ of the Clifford group $\mathcal{C}(\mathcal{H}_A)$ such that given any non-identity Pauli operators $P, Q \in \mathcal{P}(\mathcal{H}_A)$ we have that*

$$|\{C \in \mathcal{SC}(\mathcal{H}_A) | C^\dagger P C = Q\}| = \frac{|\mathcal{SC}(\mathcal{H}_A)|}{|\mathcal{P}(\mathcal{H}_A)| - 1} \quad \text{and} \quad |\mathcal{SC}(\mathcal{H}_A)| = 2^{5|A|} - 2^{3|A|}.$$

Informally, applying a random Clifford operator from $\mathcal{SC}(\mathcal{H}_A)$ (by conjugation) maps P to a Pauli operator chosen uniformly at random over all non-identity Pauli operators. Furthermore, we have that $\mathcal{P}(\mathcal{H}_A) \subset \mathcal{SC}(\mathcal{H}_A)$.

Additionally, there exists a procedure Samp *which when given as input a uniformly random string $R \leftarrow \{0,1\}^{5|A|}$ outputs in time $\mathrm{poly}(|A|)$ the classical description $\mathsf{Samp}(R)$ of a Clifford operator $C_R \in \mathcal{SC}(\mathcal{H}_A)$ with the following property: Let $U_{\mathcal{SC}(\mathcal{H}_A)}$ denote the uniform distribution over the classical descriptions of Clifford operators in $\mathcal{SC}(\mathcal{H}_A)$. Then, it holds that*

$$\mathsf{Samp}(R) \approx_{2^{-2|A|}} U_{\mathcal{SC}(\mathcal{H}_A)}, \tag{2}$$

where we recall that $\approx_{2^{-2|A|}}$ means that the statistical distance between the two distributions is at most $2^{-2|A|}$.

Pauli Twirling and Related Facts. The analysis of our construction will require the use of several facts related to Pauli twirling. We collect them below, beginning with the usual version of the Pauli twirl.

Fact 3 (Pauli twirl [34]). *Let $\rho \in \mathcal{D}(\mathcal{H}_A)$ be a state and $P, P' \in \mathcal{P}(\mathcal{H}_A)$ be Pauli operators such that $P \neq P'$. Then,*

$$\sum_{Q \in \mathcal{P}(\mathcal{H}_A)} Q^\dagger P Q \rho Q^\dagger P'^\dagger Q = 0.$$

Fact 4 (Subgroup Clifford twirl [16]). *Let $\rho \in \mathcal{D}(\mathcal{H}_A)$ be a state and $P, P' \in \mathcal{P}(\mathcal{H}_A)$ be Pauli operators such that $P \neq P'$. Let $\mathcal{SC}(\mathcal{H}_A)$ be the subgroup of Clifford group as defined in Fact 2. Then,*

$$\sum_{C \in \mathcal{SC}(\mathcal{H}_A)} C^\dagger P C \rho C^\dagger P'^\dagger C = 0.$$

As an immediate corollary, we conclude that for any normal operator $M \in \mathcal{L}(\mathcal{H}_A)$ such that $M^\dagger M = M M^\dagger$ we have that

$$\sum_{C \in \mathcal{SC}(\mathcal{H}_A)} C^\dagger P C M C^\dagger P'^\dagger C = 0,$$

since M has an eigen-decomposition.

The Transpose Method. The transpose method (see, e.g., [55]) is one of the most important tools for manipulating maximally entangled states. Note that the canonical purification of a maximally mixed state $\rho_A = U_A$, denoted $\rho_{A\hat{A}}$, is a maximally entangled state. Roughly speaking, the transpose method corresponds to the statement that some local action on one half of the maximally entangled state (say register A) is equivalent to performing the transpose of the same action on the other half of that state (register $\hat{A}$). We now state this formally.

Fact 5 (Transpose Method). *Let $\rho_{A\hat{A}}$ be the canonical purification of $\rho_A = U_A$. For any $M \in \mathcal{L}(\mathcal{H}_A)$ it holds that*

$$(M \otimes \mathbb{I}_{\hat{A}})\rho_{A\hat{A}}(M^\dagger \otimes \mathbb{I}_{\hat{A}}) = (\mathbb{I}_A \otimes M^T)\rho_{A\hat{A}}(\mathbb{I}_A \otimes (M^T)^\dagger).$$

2.3 Quantum-Secure Randomness Extractors

Randomness extractors are key objects in our constructions of non-malleable codes and secret sharing schemes. We introduce the relevant notions and auxiliary results here.

Definition 13 (Min-entropy). *Given a state ρ_{XE}, the min-entropy of X conditioned on E, denoted by $\mathrm{H}_{\min}(X\|E)_\rho$, is defined as*

$$\mathrm{H}_{\min}(X\|E)_\rho = -\inf_{\sigma_E \in \mathcal{D}(\mathcal{H}_E)} \mathrm{D}_{\max}(\rho_{XE}\|\mathbb{I}_X \otimes \sigma_E).$$

Roughly speaking, a 2-source non-malleable extractor nmExt takes as input two (not necessarily) uniformly random and independent strings X and Y and outputs a string $R = \mathrm{nmExt}(X, Y)$ that is statistically close to uniform distribution. It is called *non-malleable* because learning $\mathrm{nmExt}(f(X), g(Y))$ for any known tampering functions f and g (without fixed points) reveals essentially nothing about $R = \mathrm{nmExt}(X, Y)$, in the sense that R should still be close to uniformly random given the tampered version $\mathrm{nmExt}(f(X), g(Y))$. We may thus see (X, Y) as a form of split-state encoding of R. Since our split-state tampering adversaries have quantum capabilities *and* access to shared quantum entanglement, we cannot use an arbitrary classical 2-source non-malleable extractor to generate R. Instead, we make use of an explicit *quantum-secure* 2-source non-malleable extractor recently constructed by Batra, Boddu, and Jain [16], which remains secure against such quantum adversaries and whose properties we detail below.

Fact 6 (Quantum-Secure 2-Source Non-Malleable Extractor [16]). *Consider the split-state tampering experiment in Fig. 3 with a split-state tampering adversary $\mathcal{A} = (U, V, |\psi\rangle_{W_1W_2})$. Based on this figure, define $p_{\mathsf{same}} = \Pr[(X, Y) = (X', Y')]_{\hat{\rho}}$ and the conditioned quantum states*

$$\rho^{\mathsf{same}} = (\mathrm{nmExt} \otimes \mathrm{nmExt})(\hat{\rho}|(X, Y) = (X', Y'))$$

and

$$\rho^{\mathsf{tamp}} = (\mathrm{nmExt} \otimes \mathrm{nmExt})(\hat{\rho}|(X, Y) \neq (X', Y')).$$

For any given constant $\delta > 0$*, there exists an explicit function* $\mathrm{nmExt} : \{0,1\}^n \times \{0,1\}^{\delta n} \to \{0,1\}^r$ *with output length* $r = (1/2 - \delta)n$ *such that for independent sources* $X \leftarrow \{0,1\}^n$ *and* $Y \leftarrow \{0,1\}^{\delta n}$ *and any such split-state tampering adversary* $\mathcal{A} = (U, V, |\psi\rangle_{W_1 W_2})$ *it holds that*

1. $\|\mathrm{nmExt}(X,Y)X - U_r \otimes U_n\|_1 \leq \varepsilon$ *and* $\|\mathrm{nmExt}(X,Y)Y - U_r \otimes U_{\delta n}\|_1 \leq \varepsilon$,
2. $p_{\mathsf{same}} \|\rho^{\mathsf{same}}_{RW_2} - U_r \otimes \rho^{\mathsf{same}}_{W_2}\|_1 + (1 - p_{\mathsf{same}}) \|\rho^{\mathsf{tamp}}_{RR'W_2} - U_r \otimes \rho^{\mathsf{tamp}}_{R'W_2}\|_1 \leq \varepsilon$,

with $\varepsilon = 2^{-n^{\Omega_\delta(1)}}$*. Furthermore,* $\mathrm{nmExt}(x,y)$ *can be computed in time* $\mathrm{poly}(n)$.

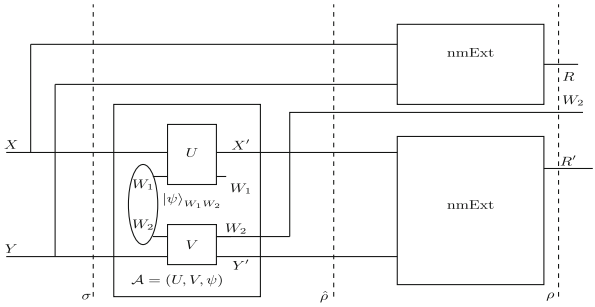

Fig. 3. Split-state tampering experiment for quantum-secure 2-source non-malleable extractors.

Intuitively, Item 1 in Fact 6 guarantees that $R = \mathrm{nmExt}(X,Y)$ remains close to uniformly random even when one of the input sources X and Y is revealed. This property is usually called *strong extraction*. Item 2 spells out the non-malleability guarantees of nmExt: If the tampering attack does not change X and Y (i.e., $(X',Y') = (X,Y)$), then R should be close to uniformly random even given one of the updated entangled states shared by the adversaries attacking each source. On the other hand, if the tampering attack changed X and Y (i.e., $(X',Y') \neq (X,Y)$), then R should be close to uniformly random even given the additional output $R' = \mathrm{nmExt}(X',Y')$ *and* one of the updated entangled states.

3 Split-State Non-malleable Codes for Quantum Messages

In this section, we describe and analyze our split-state non-malleable coding scheme for quantum messages. We begin by describing the encoding and decoding procedures Enc and Dec. In the analysis, we first show that $(\mathrm{Enc}, \mathrm{Dec})$ is an average-case non-malleable code with low error for quantum messages. This yields Theorem 1. Then, to conclude the argument and obtain Theorem 2, we show that every such average-case non-malleable code is also *worst-case* split-state non-malleable at the price of a blow-up in the error as a function of the input quantum message length.

3.1 Our Candidate Coding Scheme for Quantum Messages

We proceed to describe our explicit candidate coding scheme for quantum messages. Suppose that we wish to encode a quantum state σ_M with canonical purification $\sigma_{M\hat{M}}$. Let $b = |M|$ denote the message length and fix an arbitrary constant $\delta > 0$. We will invoke the explicit quantum-secure 2-source ε-non-malleable extractor $\mathrm{nmExt} : \{0,1\}^{\ell} \times \{0,1\}^{\delta\ell} \to \{0,1\}^{r}$ guaranteed by Fact 6 with output length r satisfying

$$r = (1/2 - \delta)\ell \geq 5b \tag{3}$$

and error $\varepsilon = 2^{-\ell^{\Omega_\delta(1)}}$, where $\Omega_\delta(\cdot)$ hides constants which depend only on δ.

The encoding CPTP map Enc works as follows on input σ_M:

1. Sample classical bitstrings $X \leftarrow \{0,1\}^{\ell}$ and $Y \leftarrow \{0,1\}^{\delta\ell}$;
2. Compute the classical key $R = \mathrm{nmExt}(X, Y) \in \{0,1\}^{r}$;
3. Let $\mathcal{SC}(\mathcal{H}_M)$ be the subgroup of the Clifford group described in Fact 2 and Samp be the associated sampling procedure. Compute $C_R = \mathsf{Samp}(R)$ and the masked state $\psi_Z = C_R \sigma_M C_R^{\dagger}$. Note that the constraint in Equation (3) guarantees that R is long enough to sample such a Clifford operator as per Fact 2.
4. Output registers X and (Y, Z) as the two parts of the split-state encoding. Note that X and Y are classical strings while Z is a quantum state.

It is clear that Enc can be computed efficiently if nmExt is explicit. The decoding procedure Dec is straightforward, and proceeds as follows on input possibly tampered registers $(X, Y) \to (X', Y')$ and $\psi_Z \to \tau_Z$:

1. Compute the candidate key $R' = \mathrm{nmExt}(X', Y')$;
2. Let τ_Z denote the possibly tampered quantum state stored in register Z. Then, compute the candidate message $\eta_M = C_{R'}^{\dagger} \tau_Z C_{R'}$ where, as in the encoding procedure Enc above, $C_{R'} = \mathsf{Samp}(R')$ with Samp the sampling procedure corresponding to the subgroup $\mathcal{SC}(\mathcal{H}_M)$ of the Clifford group from Fact 2;
3. Output η_M.

It is easy to check that this coding scheme (Enc, Dec) satisfies the basic correctness property $\mathrm{Dec}(\mathrm{Enc}(\sigma_{M\hat{M}})) = \sigma_{M\hat{M}}$.

3.2 Average-Case Non-malleability

In this section, we show that the coding scheme (Enc, Dec) described in Sect. 3.1 is average-case non-malleable with low-error. More precisely, we prove the following result.

Theorem 4 (Average-Case Non-Malleable Codes with Constant Rate). *For any fixed constant $\delta > 0$, the coding scheme* (Enc, Dec) *described in Sect. 3.1 with codewords of length n and message size $b \leq \left(\frac{1}{11} - \delta\right) n$ is average-case ε'-non-malleable with $\varepsilon' = 2^{-n^{\Omega_\delta(1)}}$. Moreover, both* Enc *and* Dec *can be computed in time* $\mathrm{poly}(n)$.

The claim about the computational complexity of Enc and Dec in Theorem 4 can be easily verified from the description in Sect. 3.1, using the fact that the codeword length n satisfies $n = O(\ell)$. We prove the remainder of Theorem 4 via a sequence of lemmas, whose proofs mostly appear in the full version of this work.

Throughout this proof we assume that C_R, the Clifford operator sampled from R via the sampling procedure Samp from Fact 2 used in Step 3 of $\mathrm{Enc}(\sigma)$ in Sect. 3.1, is uniformly distributed over $\mathcal{SC}(\mathcal{H}_M)$. Based on (2) in Fact 2, this assumption will lead to an additive factor of 2^{-2b} in the final non-malleability error, which we add at the end of our argument.

For ease of readability, a detailed diagram of the complete split-state tampering experiment on $(\mathrm{Enc}, \mathrm{Dec})$ in Fig. 4. Taking into account the notation from Fig. 4, in order to prove Theorem 4 it suffices to show that for any split-state adversary $\mathcal{A} = (U, V, \psi)$ and $\sigma_{M\hat{M}}$ maximally entangled it holds that

$$(\sigma_3)_{\hat{M}M} \approx_{\varepsilon'=2^{-n^{\Omega_\delta(1)}}} p_{\mathcal{A}}\sigma_{\hat{M}M} + (1-p_{\mathcal{A}})(U_{\hat{M}} \otimes \gamma_M^{\mathcal{A}}), \tag{4}$$

where $(p_{\mathcal{A}}, \gamma_M^{\mathcal{A}})$ depend only on the split-state adversary $\mathcal{A}$.

We begin by noting that in order to establish Eq. (4) we can equivalently consider the modified tampering experiment described in Fig. 5, where the (transposed) Clifford operator is applied to $\sigma_{\hat{M}}$ instead. More precisely, we have the following lemma.

Lemma 1. *For any fixed split-state adversary $\mathcal{A}$ it holds that the states ρ, ρ_1, ρ_2, and ρ_3 in Fig. 5 are equal to σ, σ_1, σ_2,and σ_3 in Fig. 4, respectively.*

Proof. The desired statement follows if we show that σ_1 and ρ_1 are equal. This is a direct consequence of the transpose method (Fact 5). □

We can further note that delaying the generation of R and the application of the corresponding Clifford operator C_R^T on the $\hat{M}$ register until the very end has no effect on the final state. Therefore, we can focus on the modified tampering experiment described in Fig. 6. We capture this formally in the next brief lemma statement.

Lemma 2. *For any fixed split-state adversary $\mathcal{A}$ it holds that the final state θ_4 in Fig. 6 is equal to the final state ρ_3 in Fig. 5.*

In particular, combining Lemmas 1 and 2 implies that to establish Equation (4) it suffices to show that for any split-state adversary $\mathcal{A} = (U, V, \psi)$ and $\theta_{M\hat{M}}$ maximally entangled it holds that

$$(\theta_4)_{\hat{M}M} \approx_{\varepsilon'} p_{\mathcal{A}}\theta_{\hat{M}M} + (1-p_{\mathcal{A}})(U_{\hat{M}} \otimes \gamma_M^{\mathcal{A}}), \tag{5}$$

where $(p_{\mathcal{A}}, \gamma_M^{\mathcal{A}})$ depend only on the split-state adversary $\mathcal{A}$. Here, the states $\theta, \theta_1, \theta_2, \theta_3, \theta_4$ correspond to the intermediate states of the modified tampering experiment in Fig. 6.

We now set up some helpful definitions before proceeding to the next lemma. We may write θ_2 as

$$\theta_2 = (U \otimes V)(\theta_1 \otimes |\psi\rangle_{W_1 W_2})(U \otimes V)^\dagger.$$

Our analysis will proceed by cases, depending on whether the X and Y registers are modified by the tampering experiment in Fig. 6 (i.e., $XY \neq X'Y'$) or not. To this end, we consider two different conditionings of θ_2 based on these two cases. More precisely, taking into account Definition 4, we define

$$\theta_2^{\mathsf{tamp}} = \theta_2 | (XY \neq X'Y') \quad \text{and} \quad \theta_2^{\mathsf{same}} = \theta_2 | (XY = X'Y').$$

Note that we may write $\theta_2 = p_{\mathsf{same}} \theta_2^{\mathsf{same}} + (1 - p_{\mathsf{same}}) \theta_2^{\mathsf{tamp}}$, where $p_{\mathsf{same}} = \Pr[XY = X'Y']_{\theta_2}$ is the probability that XY are not modified in the tampering experiment from Fig. 6. Further taking into account that

$$\theta_3 = (\mathrm{nmExt}_{XY} \otimes \mathrm{nmExt}_{X'Y'})\theta_2,$$

let

$$\theta_3^{\mathsf{tamp}} = (\mathrm{nmExt}_{XY} \otimes \mathrm{nmExt}_{X'Y'})\theta_2^{\mathsf{tamp}}$$

and

$$\theta_3^{\mathsf{same}} = (\mathrm{nmExt}_{XY} \otimes \mathrm{nmExt}_{X'Y'})\theta_2^{\mathsf{same}}.$$

Let $D_{R\hat{M}}$ denote the controlled Clifford operator C_R^T acting on register $\hat{M}$. Similarly, let $\tilde{D}_{R'Z}$ denote the controlled Clifford operator $C_{R'}^\dagger$ acting on register Z. We can then write

$$(\theta_4)_{\hat{M}M} = (D_{R\hat{Z}} \otimes \tilde{D}_{R'Z})(\theta_3)_{RR'Z\hat{M}}(D_{R\hat{Z}}^\dagger \otimes \tilde{D}_{R'Z}^\dagger).$$

Analogously to the previous cases, we define

$$(\theta_4^{\mathsf{tamp}})_{\hat{M}M} = (D_{R\hat{Z}} \otimes \tilde{D}_{R'Z})(\theta_3^{\mathsf{tamp}})_{RR'Z\hat{M}}(D_{R\hat{Z}}^\dagger \otimes \tilde{D}_{R'Z}^\dagger) \tag{6}$$

and

$$(\theta_4^{\mathsf{same}})_{\hat{M}M} = (D_{R\hat{Z}} \otimes \tilde{D}_{R'Z})(\theta_3^{\mathsf{same}})_{RR'Z\hat{M}}(D_{R\hat{Z}}^\dagger \otimes \tilde{D}_{R'Z}^\dagger). \tag{7}$$

Recall from Equation (5) that, based on Lemmas 1 and 2, our end goal is to show that

$$(\theta_4)_{\hat{M}M} \approx_{\varepsilon'} p_{\mathcal{A}} \theta_{\hat{M}M} + (1 - p_{\mathcal{A}})(U_{\hat{M}} \otimes \gamma_M^{\mathcal{A}})$$

for some quantity $p_{\mathcal{A}} \in [0, 1]$ and state $\gamma_M^{\mathcal{A}}$ depending only on the split-state adversary $\mathcal{A}$. Towards establishing this, we begin by writing

$$(\theta_4)_{\hat{M}M} = p_{\mathsf{same}}(\theta_4^{\mathsf{same}})_{\hat{M}M} + (1 - p_{\mathsf{same}})(\theta_4^{\mathsf{tamp}})_{\hat{M}M}.$$

We will focus on each of the two terms on the right-hand side of this equation separately. We invoke the two lemmas below, whose proofs we defer to later dedicated sections.

The first lemma is relevant for handling θ_4^{tamp}, and intuitively states that when tampering occurs and $X'Y' \neq XY$ the outcome of the tampering experiment is close (in trace distance) to being an unentangled message.

Lemma 3. *We have that* $(1 - p_{\mathsf{same}}) \left\| (\theta_4^{\mathsf{tamp}})_{\hat{M}M} - U_{\hat{M}} \otimes (\theta_4^{\mathsf{tamp}})_M \right\|_1 \leq \varepsilon$.

Proof. See the full version. □

The second lemma is relevant for handling θ_4^{same}, and intuitively states that when $X'Y' = XY$ the outcome of the tampering experiment is close (in trace distance) to being either the original message or a completely independent and unentangled message.

Lemma 4. *There exists a constant* $p_{\mathsf{epr}} \in [0,1]$ *depending only on the split-state adversary* $\mathcal{A}$ *such that*

$$p_{\mathsf{same}} \cdot \left\| (\theta_4^{\mathsf{same}})_{\hat{M}M} - \left(p_{\mathsf{epr}} \cdot \theta_{\hat{M}M} + (1 - p_{\mathsf{epr}})(U_{\hat{M}} \otimes U_M) \right) \right\|_1 \leq \varepsilon + 2 \cdot 4^{-b}.$$

Proof. See the full version. □

We now show how to wrap up the argument and establish Eq. (5) using Lemmas 3 and 4. We have

$$\begin{aligned}
&(\theta_4)_{\hat{M}M} \\
&= p_{\mathsf{same}} (\theta_4^{\mathsf{same}})_{\hat{M}M} + (1 - p_{\mathsf{same}})(\theta_4^{\mathsf{tamp}})_{\hat{M}M} \\
&\approx_{\varepsilon + 2 \cdot 4^{-b}} p_{\mathsf{same}} \left(p_{\mathsf{epr}} \theta_{M\hat{M}} + (1 - p_{\mathsf{epr}})(U_{\hat{M}} \otimes U_M) \right) + (1 - p_{\mathsf{same}})(\theta_4^{\mathsf{tamp}})_{\hat{M}M} \quad (8) \\
&\approx_{\varepsilon} p_{\mathsf{same}} \left(p_{\mathsf{epr}} \cdot \theta_{M\hat{M}} + (1 - p_{\mathsf{epr}})(U_{\hat{M}} \otimes U_M) \right) + (1 - p_{\mathsf{same}})(U_{\hat{M}} \otimes (\theta_4^{\mathsf{tamp}})_M), \quad (9)
\end{aligned}$$

where Eq. (8) follows from Lemma 4 and Equation (9) follows from Lemma 3. Combining Eqs. (8) and (9) with a triangle inequality shows that

$$\begin{aligned}
&(\theta_4)_{\hat{M}M} \\
&\approx_{2(\varepsilon + 4^{-b})} p_{\mathsf{same}} \left(p_{\mathsf{epr}} \cdot \theta_{M\hat{M}} + (1 - p_{\mathsf{epr}})(U_{\hat{M}} \otimes U_M) \right) \\
&+ (1 - p_{\mathsf{same}})(U_{\hat{M}} \otimes (\theta_4^{\mathsf{tamp}})_M) \\
&= p_{\mathsf{same}} \cdot p_{\mathsf{epr}} \cdot \theta_{M\hat{M}} \\
&+ (1 - p_{\mathsf{same}} \cdot p_{\mathsf{epr}}) \left(U_{\hat{M}} \otimes \left(\frac{p_{\mathsf{same}}(1 - p_{\mathsf{epr}})}{1 - p_{\mathsf{same}} \cdot p_{\mathsf{epr}}} \cdot U_M + \frac{1 - p_{\mathsf{same}}}{1 - p_{\mathsf{same}} \cdot p_{\mathsf{epr}}} \cdot (\theta_4^{\mathsf{tamp}})_M \right) \right). \quad (10)
\end{aligned}$$

Consider setting $p_{\mathcal{A}} = p_{\mathsf{same}} \cdot p_{\mathsf{epr}}$ and $\gamma_M^{\mathcal{A}} = \frac{p_{\mathsf{same}}(1 - p_{\mathsf{epr}})}{1 - p_{\mathsf{same}} \cdot p_{\mathsf{epr}}} \cdot U_M + \frac{1 - p_{\mathsf{same}}}{1 - p_{\mathsf{same}} \cdot p_{\mathsf{epr}}} \cdot (\theta_4^{\mathsf{tamp}})_M$ in Equation (10). We claim that $p_{\mathcal{A}}$ and $\gamma_M^{\mathcal{A}}$ depend only on the split-state adversary $\mathcal{A}$. This holds because:

- $p_{\mathsf{same}} = \Pr[XY = X'Y']_{\theta_2}$ is a function of $\mathcal{A}$ and (X, Y), which are sampled independently of the message σ_M;
- Lemma 4 guarantees that p_{epr} is a function of $\mathcal{A}$ only;
- The state $(\theta_4^{\mathsf{tamp}})_M$ is a function of $(\theta_3^{\mathsf{tamp}})_{ZX'Y'}$, and one can prepare the latter by running an independent tampering experiment on registers $XYZ = U_l \otimes U_{\delta l} \otimes U_Z$.

Therefore, we conclude that both $p_{\mathcal{A}}$ and $\gamma_M^{\mathcal{A}}$ only depend on $\mathcal{A}$. In this case, we can write

$$(\theta_4)_{\hat{M}M} \approx_{2(\varepsilon+4^{-b})} p_{\mathcal{A}} \cdot \theta_{M\hat{M}} + (1-p_{\mathcal{A}}) \left(U_{\hat{M}} \otimes \gamma_M^{\mathcal{A}}\right).$$

By Lemmas 1 and 2, this implies that we have

$$(\sigma_3)_{\hat{M}M} \approx_{2(\varepsilon+4^{-b})} p_{\mathcal{A}} \cdot \sigma_{M\hat{M}} + (1-p_{\mathcal{A}}) \left(U_{\hat{M}} \otimes \gamma_M^{\mathcal{A}}\right) \tag{11}$$

in the original tampering experiment from Fig. 4.

Setting Parameters for Theorem 4. To conclude the proof of Theorem 4, it remains to argue that, given any constant $\delta > 0$, the coding scheme $(\mathsf{Enc}, \mathsf{Dec})$ is average-case $(\varepsilon' = 2^{-n^{\Omega_\delta(1)}})$-non-malleable for any message length $b \leq (\frac{1}{11} - \delta)n$. This is done by setting parameters in the construction above appropriately, which can be found in the full version.

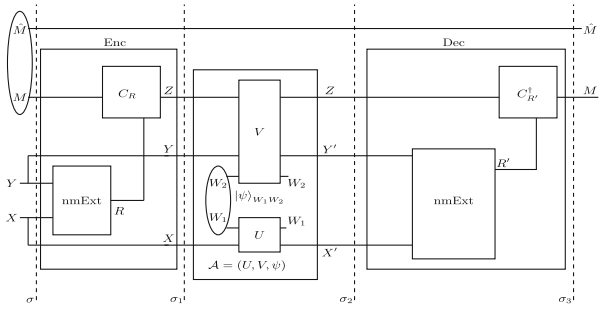

Fig. 4. Split-state tampering experiment.

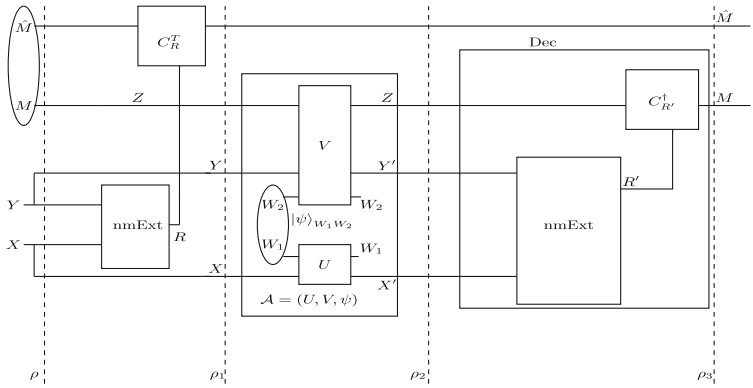

Fig. 5. Split-state tampering experiment after applying the transpose method.

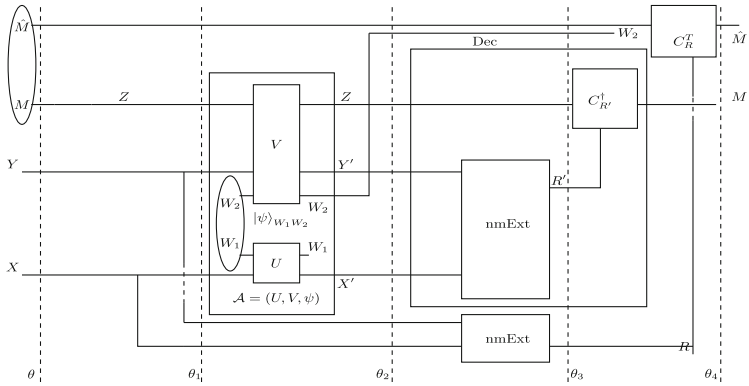

Fig. 6. Split-state tampering experiment after applying the transpose method and delaying both the generation of register R and the application of the corresponding Clifford operator.

3.3 From Average-Case to Worst-Case Non-malleability

In this section, we show that every average-case non-malleable code is also worst-case non-malleable with larger error, provided that the message length is not too large. More precisely, we have the following.

Lemma 5. *If* $(\mathrm{Enc}, \mathrm{Dec})$ *is an average-case* ε*-non-malleable code for quantum messages of length* b*, then it is also a (worst-case)* ε'*-non-malleable code for quantum messages of length* b*, where* $\varepsilon' = 2^b \cdot \varepsilon$*.*

Proof. See the full version. □

Combining Theorem 4 with Lemma 5 immediately implies the following.

Theorem 5. *There exists a constant* $c \in (0, 1)$ *such that the following holds: The coding scheme* $(\mathrm{Enc}, \mathrm{Dec})$ *described in Sect. 3.1 with codewords of length* n *and any message length* $b \leq n^c$ *is (worst-case)* ε*-non-malleable with* $\varepsilon = 2^{-n^{\Omega(1)}}$*. Moreover, both* Enc *and* Dec *can be computed in time* $\mathrm{poly}(n)$*.*

Acknowledgements. We thank Thiago Bergamaschi for insightful discussions about notions of non-malleability in the quantum setting. We also thank Dakshita Khurana for useful discussions in the initial stage of this project.

JR was supported in part by NOVA LINCS (ref. UIDB/04516/2020) with the financial support of FCT - Fundação para a Ciência e a Tecnologia. RJ was supported by the NRF grant NRF2021-QEP2-02-P05 and the Ministry of Education, Singapore, under the Research Centres of Excellence program. This work was done in part while RJ was visiting the Technion-Israel Institute of Technology, Haifa, Israel and the Simons Institute for the Theory of Computing, Berkeley, CA, USA.

References

1. Aggarwal, D., Agrawal, S., Gupta, D., Maji, H.K., Pandey, O., Prabhakaran, M.: Optimal computational split-state non-malleable codes. In: Kushilevitz, E., Malkin, T. (eds.) Theory of Cryptography, pp. 393–417. Springer, Heidelberg (2016)
2. Aggarwal, D., Boddu, N.G., Jain, R.: Quantum secure non-malleable codes in the split-state model. IEEE Trans. Inf. Theory **70**(1), 349–371 (2024). https://doi.org/10.1109/TIT.2023.3328839
3. Aggarwal, D., et al.: Stronger leakage-resilient and non-malleable secret sharing schemes for general access structures. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019. LNCS, vol. 11693, pp. 510–539. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26951-7_18
4. Aggarwal, D., Dodis, Y., Lovett, S.: Non-malleable codes from additive combinatorics. SIAM J. Comput. **47**(2), 524–546 (2018). https://doi.org/10.1137/140985251. Preliminary version in STOC 2014
5. Aggarwal, D., Dziembowski, S., Kazana, T., Obremski, M.: Leakage-resilient non-malleable codes. In: Dodis, Y., Nielsen, J.B. (eds.) Theory of Cryptography, pp. 398–426. Springer, Heidelberg (2015)
6. Aggarwal, D., Kanukurthi, B., Obbattu, S.L.B., Obremski, M., Sekar, S.: Rate one-third non-malleable codes. In: Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing (STOC 2022), pp. 1364–1377. Association for Computing Machinery, New York (2022). https://doi.org/10.1145/3519935.3519972
7. Aggarwal, D., Obremski, M.: A constant rate non-malleable code in the split-state model. In: 2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS), pp. 1285–1294. IEEE Computer Society, Los Alamitos (2020). https://doi.org/10.1109/FOCS46700.2020.00122
8. Alagic, G., Majenz, C.: Quantum non-malleability and authentication. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017. LNCS, vol. 10402, pp. 310–341. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63715-0_11
9. Ambainis, A., Bouda, J., Winter, A.: Nonmalleable encryption of quantum information. J. Math. Phys. **50**(4), 042106 (2009). https://doi.org/10.1063/1.3094756
10. Badrinarayanan, S., Srinivasan, A.: Revisiting non-malleable secret sharing. In: Ishai, Y., Rijmen, V. (eds.) EUROCRYPT 2019. LNCS, vol. 11476, pp. 593–622. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-17653-2_20
11. Ball, M., Chattopadhyay, E., Liao, J., Malkin, T., Tan, L.: Non-malleability against polynomial tampering. In: Micciancio, D., Ristenpart, T. (eds.) Advances in Cryptology - CRYPTO 2020, pp. 97–126. Springer (2020).https://doi.org/10.1007/978-3-030-56877-1_4
12. Ball, M., Dachman-Soled, D., Kulkarni, M., Malkin, T.: Non-malleable codes for bounded depth, bounded fan-in circuits. In: Fischlin, M., Coron, J.-S. (eds.) EUROCRYPT 2016. LNCS, vol. 9666, pp. 881–908. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-49896-5_31
13. Ball, M., Dachman-Soled, D., Kulkarni, M., Malkin, T.: Non-malleable codes from average-case hardness: AC^0, decision trees, and streaming space-bounded tampering. In: Nielsen, J.B., Rijmen, V. (eds.) Advances in Cryptology - EUROCRYPT 2018, pp. 618–650. Springer (2018).https://doi.org/10.1007/978-3-319-78372-7_20
14. Ball, M., Guo, S., Wichs, D.: Non-malleable codes for decision trees. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019. LNCS, vol. 11692, pp. 413–434. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26948-7_15

15. Barnum, H., Crepeau, C., Gottesman, D., Smith, A., Tapp, A.: Authentication of quantum messages. In: The 43rd Annual IEEE Symposium on Foundations of Computer Science (FOCS), 2002, pp. 449–458 (2002). https://doi.org/10.1109/SFCS.2002.1181969
16. Batra, R., Boddu, N.G., Jain, R.: Quantum secure non-malleable randomness encoder and its applications. arXiv preprint arXiv:2308.07340 (2023). Contributed talk at QCRYPT 2023
17. Bergamaschi, T.: Pauli manipulation detection codes and applications to quantum communication over adversarial channels. In: Joye, M., Leander, G. (eds.) Advances in Cryptology – EUROCRYPT 2024, pp. 404–433. Springer, Cham (2024). https://arxiv.org/abs/2304.06269
18. Blakley, G.R.: Safeguarding cryptographic keys. In: 1979 International Workshop on Managing Requirements Knowledge (MARK), pp. 313–318 (1979). https://doi.org/10.1109/MARK.1979.8817296
19. Boddu, N.G., Goyal, V., Jain, R., Ribeiro, J.: Split-state non-malleable codes and secret sharing schemes for quantum messages. arXiv preprint arXiv:2308.06466
20. Boddu, N.G., Jain, R., Kapshikar, U.: Quantum secure non-malleable-extractors. arXiv preprint arXiv:2109.03097 (2021). Contributed talk at TQC 2022
21. Brian, G., Faonio, A., Obremski, M., Simkin, M., Venturi, D.: Non-malleable secret sharing against bounded joint-tampering attacks in the plain model. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020. LNCS, vol. 12172, pp. 127–155. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-56877-1_5
22. Brian, G., Faonio, A., Ribeiro, J., Venturi, D.: Short non-malleable codes from related-key secure block ciphers, revisited. IACR Trans. Symmet. Cryptol. **2022**(3), 1–19 (2022). https://doi.org/10.46586/tosc.v2022.i3.1-19
23. Brian, G., Faonio, A., Venturi, D.: Continuously non-malleable secret sharing: joint tampering, plain model and capacity. In: Nissim, K., Waters, B. (eds.) Theory of Cryptography, pp. 333–364. Springer, Cham (2021)
24. Brian, G., Faust, S., Micheli, E., Venturi, D.: Continuously non-malleable codes against bounded-depth tampering. In: Agrawal, S., Lin, D. (eds.) Advances in Cryptology - ASIACRYPT 2022, pp. 384–413. Springer (2022). https://doi.org/10.1007/978-3-031-22972-5_14
25. Broadbent, A., Wainewright, E.: Efficient simulation for quantum message authentication. In: Nascimento, A.C., Barreto, P. (eds.) Information Theoretic Security, pp. 72–91. Springer, Cham (2016)
26. Çakan, A., Goyal, V., Liu-Zhang, C.D., Ribeiro, J.: Unbounded leakage-resilience and intrusion-detection in a quantum world (2024). https://eprint.iacr.org/2023/410, to appear at TCC 2024. Contributed talk at TQC 2024. https://eprint.iacr.org/2023/410
27. Chandran, N., Kanukurthi, B., Obbattu, S.L.B., Sekar, S.: Short leakage resilient and non-malleable secret sharing schemes. In: Dodis, Y., Shrimpton, T. (eds.) Advances in Cryptology - CRYPTO 2022, pp. 178–207. Springer (2022). https://doi.org/10.1007/978-3-031-15802-5_7
28. Chattopadhyay, E., Goyal, V., Li, X.: Nonmalleable extractors and codes, with their many tampered extensions. SIAM J. Comput. **49**(5), 999–1040 (2020). https://doi.org/10.1137/18M1176622. Preliminary version in STOC 2016
29. Cheraghchi, M., Guruswami, V.: Non-malleable coding against bit-wise and split-state tampering. In: Proceedings of Theory of Cryptography Conference (TCC), pp. 440–464 (2014). https://doi.org/10.1007/978-3-642-54242-8_19. Extended Version in Journal of Cryptology

30. Cheraghchi, M., Guruswami, V.: Capacity of non-malleable codes. IEEE Trans. Inf. Theory **62**(3), 1097–1118 (2016). https://doi.org/10.1109/TIT.2015.2511784
31. Clauser, J.F., Horne, M.A., Shimony, A., Holt, R.A.: Proposed experiment to test local hidden-variable theories. Phys. Rev. Lett. **23**, 880–884 (1969). https://doi.org/10.1103/PhysRevLett.23.880
32. Cleve, R., Gottesman, D., Lo, H.K.: How to share a quantum secret. Phys. Rev. Lett. **83**, 648–651 (1999). https://doi.org/10.1103/PhysRevLett.83.648
33. Cleve, R., Leung, D., Liu, L., Wang, C.: Near-linear constructions of exact unitary 2-designs. Quantum Info. Comput. **16**(9–10), 721–756 (2016)
34. Dankert, C., Cleve, R., Emerson, J., Livine, E.: Exact and approximate unitary 2-designs and their application to fidelity estimation. Phys. Rev. A **80**, 012304 (2009). https://doi.org/10.1103/PhysRevA.80.012304
35. Datta, N.: Min- and max- relative entropies and a new entanglement monotone. IEEE Trans. Inf. Theory **55**, 2816–2826 (2009)
36. Dziembowski, S., Kazana, T., Obremski, M.: Non-malleable codes from two-source extractors. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013. LNCS, vol. 8043, pp. 239–257. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40084-1_14
37. Dziembowski, S., Pietrzak, K., Wichs, D.: Non-malleable codes. J. ACM **65**(4), 1–32 (2018). https://doi.org/10.1145/3178432. Preliminary version in ICS 2010
38. Faonio, A., Venturi, D.: Non-malleable secret sharing in the computational setting: adaptive tampering, noisy-leakage resilience, and improved rate. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019. LNCS, vol. 11693, pp. 448–479. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26951-7_16
39. Faust, S., Hostáková, K., Mukherjee, P., Venturi, D.: Non-malleable codes for space-bounded tampering. In: Katz, J., Shacham, H. (eds.) Advances in Cryptology - CRYPTO 2017, pp. 95–126. Springer (2017). https://doi.org/10.1007/978-3-319-63715-0_4
40. Faust, S., Mukherjee, P., Nielsen, J.B., Venturi, D.: Continuous non-malleable codes. In: Lindell, Y. (ed.) Theory of Cryptography - 11th Theory of Cryptography Conference, TCC 2014, pp. 465–488. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-54242-8_20
41. Fehr, S., Karpman, P., Mennink, B.: Short non-malleable codes from related-key secure block ciphers. IACR Trans. Symmet. Cryptol. **2018**(1), 336–352 (2018). https://doi.org/10.13154/tosc.v2018.i1.336-352
42. Goyal, V., Kumar, A.: Non-malleable secret sharing. In: Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing (STOC 2018), pp. 685–698. Association for Computing Machinery, New York (2018). https://doi.org/10.1145/3188745.3188872
43. Goyal, V., Kumar, A.: Non-malleable secret sharing for general access structures. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018. LNCS, vol. 10991, pp. 501–530. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-96884-1_17
44. Goyal, V., Pandey, O., Richelson, S.: Textbook non-malleable commitments. In: Proceedings of the Forty-Eighth Annual ACM Symposium on Theory of Computing (STOC 2016), pp. 1128–1141. Association for Computing Machinery, New York (2016). https://doi.org/10.1145/2897518.2897657
45. Goyal, V., Srinivasan, A., Zhu, C.: Multi-source non-malleable extractors and applications. In: Canteaut, A., Standaert, F.-X. (eds.) EUROCRYPT 2021. LNCS, vol. 12697, pp. 468–497. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-77886-6_16

46. Jain, R., Radhakrishnan, J., Sen, P.: Privacy and interaction in quantum communication complexity and a theorem about the relative entropy of quantum states. In: The 43rd Annual IEEE Symposium on Foundations of Computer Science, 2002, pp. 429–438 (2002). https://doi.org/10.1109/SFCS.2002.1181967
47. Kanukurthi, B., Obbattu, S.L.B., Sekar, S.: Non-malleable randomness encoders and their applications. In: Nielsen, J.B., Rijmen, V. (eds.) Advances in Cryptology - EUROCRYPT 2018, pp. 589–617. Springer, Cham (2018)
48. Kiayias, A., Liu, F.H., Tselekounis, Y.: Practical non-malleable codes from ℓ-more extractable hash functions. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS 2016), pp. 1317–1328. Association for Computing Machinery, New York (2016). https://doi.org/10.1145/2976749.2978352
49. Li, X.: Three-source extractors for polylogarithmic min-entropy. In: 2015 IEEE 56th Annual Symposium on Foundations of Computer Science, pp. 863–882 (2015). https://doi.org/10.1109/FOCS.2015.58
50. Li, X.: Improved non-malleable extractors, non-malleable codes and independent source extractors. In: Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing (STOC 2017), pp. 1144–1156. Association for Computing Machinery, New York (2017). https://doi.org/10.1145/3055399.3055486
51. Li, X.: Non-malleable extractors and non-malleable codes: partially optimal constructions. In: Proceedings of the 34th Computational Complexity Conference (CCC 2019). Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, DEU (2019). https://doi.org/10.4230/LIPIcs.CCC.2019.28
52. Li, X.: Two source extractors for asymptotically optimal entropy, and (many) more. In: 2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS), pp. 1271–1281 (2023). https://doi.org/10.1109/FOCS57990.2023.00075
53. Liu, F.-H., Lysyanskaya, A.: Tamper and leakage resilience in the split-state model. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 517–532. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-32009-5_30
54. Nielsen, M.A., Chuang, I.L.: Quantum Computation and Quantum Information. Cambridge University Press, Cambridge (2000)
55. Ozols, M.: Lecture 5: quantum information processing protocols, quantum computing: exercise sheet 2 (2016). https://www.cl.cam.ac.uk/teaching/1617/QuantComp/exercise2.pdf
56. Shamir, A.: How to share a secret. Commun. ACM **22**(11), 612–613 (1979). https://doi.org/10.1145/359168.359176
57. Srinivasan, A., Vasudevan, P.N.: Leakage resilient secret sharing and applications. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019. LNCS, vol. 11693, pp. 480–509. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26951-7_17
58. Watrous, J.: The Theory of Quantum Information. Cambridge University Press, Cambridge (2018)

Cryptography in the Common Haar State Model: Feasibility Results and Separations

Prabhanjan Ananth(✉), Aditya Gulati, and Yao-Ting Lin

UCSB, Santa Barbara, USA
prabhanjan@cs.ucsb.edu, {adityagulati,yao-ting_lin}@ucsb.edu

Abstract. Common random string model is a popular model in classical cryptography. We study a quantum analogue of this model called the common Haar state (CHS) model. In this model, every party participating in the cryptographic system receives many copies of one or more i.i.d Haar random states.

We study feasibility and limitations of cryptographic primitives in this model and its variants:

- We present a construction of pseudorandom function-like states with security against computationally unbounded adversaries, as long as the adversaries only receive (a priori) bounded number of copies. By suitably instantiating the CHS model, we obtain a new approach to construct pseudorandom function-like states in the plain model.
- We present separations between pseudorandom function-like states (with super-logarithmic length) and quantum cryptographic primitives, such as interactive key agreement and bit commitment, with classical communication. To show these separations, we prove new results on the indistinguishability of identical versus independent Haar states against LOCC (local operations, classical communication) adversaries.

1 Introduction

In classical cryptography, the common random string and the common reference string models were primarily introduced to tackle cryptographic tasks that were impossible to achieve in the plain model. In the common reference string model, there is a trusted setup who produces a string that every party has access to. In the common random string model, the common string available to all the parties is sampled uniformly at random. Due to the lack of structure required from the common random string model, it is in general the more desirable model of the two. There have been many constructions proposed over the years in these two models, including non-interactive zero-knowledge [BFM19], secure computation with universal composition [CF01, CLOS02] and two-round secure computation [GS22, BL18].

It is a worthy pursuit to study similar models for quantum cryptographic protocols. In the quantum world, there is an option to define models that are intrinsically quantum in nature. For instance, we could define a model wherein

E. Boyle and M. Mahmoody (Eds.): TCC 2024, LNCS 15365, pp. 94–125, 2025.
https://doi.org/10.1007/978-3-031-78017-2_4

a trusted setup produces a quantum state and every party participating in the cryptographic system receives one or more copies of this quantum state. Indeed, two works by Morimae, Nehoran and Yamakawa [MNY23] and Qian [Qia23] consider this model, termed as the *common reference quantum state model* (CRQS). They proposed a construction of unconditionally secure commitments in this model. Quantum commitments is a foundational notion in quantum cryptography. In recent years, quantum commitments have been extensively studied [AQY22, MY21, AGQY22, MY23, BCQ23, Bra23] due to its implication to secure computation [BCKM21, GLSV21]. The fact that information-theoretically secure commitments are impossible in the plain model [LC97, May97, CLM23] renders the contributions of [MNY23, Qia23] particularly interesting.

Common Haar State Model. While CRQS is a quantum analogue of the common reference string model, in a similar vein, we can ask if there is a quantum analogue of the common random string model. We consider a novel model called the *common Haar state model* (CHS). In this model, every party in the system (including the adversary) receives many copies of many i.i.d Haar states. We believe that the CHS model is more pragmatic than the CRQS model owing to the fact that we do not require any structure from the common public state. This raises the possibility of avoiding a trusted setup altogether and instead we could rely upon naturally occuring physical processes to obtain the Haar states. This model was also recently introduced in an independent and concurrent recent work by Chen, Coladangelo and Sattath [CCS24] (henceforth, referred to as CCS).

There are three reasons to study this model. Firstly, this model allows us to bypass impossibility results in the plain model. For instance, as we will see later, primitives that require computational assumptions in the plain model, can instead be designed with information-theoretic security in the CHS model. Second, perhaps a less intuitive reason, is that the constructions proposed in this model can, in some cases, be adopted to obtain constructions in the plain model by instantiating the Haar states either using state designs or pseudorandom state generators (PRSGs) [JLS18]. This leads to a modular approach of designing cryptographic primitives from PRS: first design the primitive in the CHS model and then instantiate the common Haar state using PRS. Finally, this model can be leveraged to demonstrate separations between different quantum cryptographic primitives.

1.1 Our Results

We explore both feasibility results and black-box separations in the CHS model.

Feasibility Results.

Pseudorandom Function-like States with Statistical Security. We study the possibility of designing pseudorandom function-like state generators (PRFSGs), introduced by Ananth, Qian and Yuen [AQY22], with statistical security in the CHS

model. Roughly speaking, a PRFSG is an efficient keyed quantum circuit that can be used to produce many pseudorandom states. We refer the reader to Appendix A of the full version [AGL24] for a detailed discussion on the different notions of pseudorandomness in the quantum world.

We are interested in designing (λ, m, n, t)-PRFSGs in the setting when $n \geq \lambda$ and $m = \Omega(\log(\lambda))$, where λ is the key length, m is the input length, n is the output length (and also the number of the qubits in the common Haar state) and t is the maximum number of queries that can be requested by the adversary. However, in the CHS model, we can in fact achieve statistical security.

We show the following.

Theorem 1 (Informal). *There is a statistically secure (λ, m, n, ℓ)-PRFSG in the CHS model, for $m = \lambda^c$, $n \geq \lambda$ and $\ell = O\left(\frac{\lambda^{1-c}}{\log(\lambda)^{1+\varepsilon}}\right)$, for any constant $\varepsilon > 0$ and for all $c \in [0, 1)$.*

CCS is the only other work that has studied pseudorandomness in the CHS model. There are a few advantages of our result over CCS:

- Our theorem subsumes and generalizes the result of CCS who showed (λ, n, t)-PRSGs exists in their model, where the output length is larger than the key length, i.e., $n > \lambda$ and moreover, when $t = 1$ with t being the number of copies of the PRS state given to the adversary.
- Our construction, when restricted to the case of PRSGs, is slightly simpler than CCS: in CCS, on a subset of qubits of the Haar state, a random Pauli operator is applied whereas in our case a random Pauli Z operator is applied. Our construction of PRFSG uses the seminal Goldreich-Goldwasser-Micali approach [GGM86] to go from one-query security to many-query security.
- They propose novel sophisticated tools in their analysis whereas our analysis is arguably more elementary using well known facts about symmetric subspaces.
- Finally, we can achieve arbitrary stretch whereas it is unclear whether this is also achieved by CCS.

As a side contribution, the proof of our PRSG construction also simplifies the proof of the quantum public-key construction of Coladangelo [Col23]; this is due to the fact the core lemma proven in [Col23] is implied by the above theorem.

Interestingly, the above theorem has implications for computationally secure pseudorandomness in the plain model. Specifically, we obtain the following corollary by instantiating the CHS model using stretch PRSGs:

Corollary 1. Assuming (λ, n, ℓ)-PRSGs, there exists (λ', m, n, t)-PRFSGs, where $n > \lambda' > \lambda$, $m = \lambda^c$ and $\ell = O\left(\frac{\lambda^{1-c}}{\log(\lambda)^{1+\varepsilon}}\right)$, for any constant $\varepsilon > 0$ and $c \in [0, 1)$.

Prior to our work, stretch PRFSGs for super-logarithmic input length, even in the bounded query setting, was only known from one-way functions [AQY22]. This complements the work of [AQY22] who showed a construction of PRFSGs for logarithmic input length from PRSGs.

Interestingly, the state generators in both works (CCS and ours) only consume one copy of a single Haar state. In this special case, it is interesting to understand whether we can extend our result to the setting when the adversary receives $\frac{\lambda}{\log(\lambda)}$ copies or more. We show this is not possible.

Theorem 2 (Informal). *There does not exist a secure (λ, m, n, ℓ)-PRFSG, for any $m \geq 1$, in the CHS model, where $n = \omega(\log(\lambda))$ and $\ell = \Omega\left(\frac{\lambda}{\log(\lambda)}\right)$.*

CCS also proved a lower bound where they showed that unbounded copy pseudorandom states do not exist. Their negative result is stronger in the sense that they rule out PRSGs who use up many copies of the Haar states from the CHRS and thus, their work gives a clean separation between 1-copy stretch PRS and unbounded copy PRS which was not known before. On the other hand, for the special case when the PRFSG takes only one copy of the Haar state, we believe our result yields better parameters.

Commitments. In addition to pseudorandomness, we also study the possibility of constructing other cryptographic primitives in the CHS model. We show the following:

Theorem 3 (Informal). *There is an unconditionally secure bit commitment scheme in the CHS model.*

Both our construction and the commitments scheme proposed by CCS are different although they share strong similarities.

Black-Box Separations

LOCC Indistinguishability. We separate pseudorandom function-like states and quantum cryptographic primitives with classical communication using a variant of the CHS model. At the heart of our separations is a novel result that proves indistinguishability of identical versus independent Haar states against LOCC (local operations, classical communication) adversaries. More precisely, (A, B) is an LOCC adversary if A and B are quantum algorithms who can communicate with each other via only classical communication channels. It is important that A and B do not share any entanglement. Moreover, we restrict our attention to LOCC distinguishers which are LOCC adversaries of the form (A, B) where A does not output anything whereas B outputs a single bit. We say that a LOCC distinguisher (A, B) can distinguish two states ρ_{AB} and σ_{AB} with probability at most ε, referred to as ε-LOCC indistinguishability, where A receives the register A and B receives the register B, if $|\mathsf{Pr}[1 \leftarrow (A, B)(\rho_{\mathsf{AB}})] - \mathsf{Pr}[1 \leftarrow (A, B)(\sigma_{\mathsf{AB}})]| = \varepsilon$. Of particular interest is the case when

$$\rho_{\mathsf{AB}} = \mathop{\mathbb{E}}_{|\psi\rangle \leftarrow \mathcal{H}_n} \left[(|\psi\rangle^{\otimes t})_{\mathsf{A}} \otimes (|\psi\rangle^{\otimes t})_{\mathsf{B}} \right], \ \sigma_{\mathsf{AB}} = \mathop{\mathbb{E}}_{\substack{|\psi\rangle \leftarrow \mathcal{H}_n, \\ |\phi\rangle \leftarrow \mathcal{H}_n}} \left[(|\psi\rangle^{\otimes t})_{\mathsf{A}} \otimes (|\phi\rangle^{\otimes t})_{\mathsf{B}} \right]$$

Here, $\mathcal{H}_n$ denotes the Haar distribution on n-qubit quantum states and t is polynomial in n. A couple of works by Harrow [Har23] and Chen, Cotler, Huang and Li [CCHL22] prove that the LOCC indistinguishability of ρ_{AB} and σ_{AB} is negligible in n in the case when $t = 1$. In this work, we extend to the case when t is arbitrary.

Theorem 4. *ρ_{AB} and σ_{AB} (defined above) are ε-LOCC indistinguishable, where $\varepsilon = O\left(\frac{t^2}{2^n}\right)$.*

We also show that the above bound is tight by demonstrating an LOCC distinguisher whose distinguishing probability is $\Theta(\frac{t^2}{2^n})$.

Recently, Ananth, Kaleoglu and Yuen [AKY24] prove the indistinguishability of ρ_{AB} and σ_{AB} in the dual setting, against non-local adversaries that can share entanglement but cannot communicate.

The above theorem can easily be extended to the multi-party setting where either all the parties get (many copies of) the same Haar state or they receive i.i.d Haar states.

Separations. We use Theorem 4 to show that some quantum cryptographic primitives with classical communication are impossible in the CHS model. Let us develop some intuition towards proving such a statement. Suppose there are two or more parties participating in a quantum cryptographic protocol with classical communication in the CHS model. By definition, all the parties would receive many, say t, copies of $|\psi\rangle$, where $|\psi\rangle$ is sampled from the Haar distribution. Since the parties can only exchange classical messages, thanks to Theorem 4, without affecting correctness or security we can modify the protocol wherein for each party, say P_i, a Haar state $|\psi_i\rangle$ is sampled and t copies of $|\psi_i\rangle$ is given to P_i. From this, we can extract a quantum cryptographic primitive in the plain model since each party can sample a Haar state on its own. In conclusion, quantum cryptographic primitives with classical communication in the CHS model can be turned into their counterparts in the plain model.

This gives a natural recipe for proving impossibility results in the CHS model. We apply this recipe to obtain impossibility results for interactive key agreements and interactive commitments.

Theorem 5. *Interactive quantum key agreement and interactive quantum commitment protocols, with classical communication, are impossible in the CHS model.*

We extend the above theorem to separate interactive quantum key agreement and interactive quantum commitments from pseudorandom function-like state generators. The separations are obtained by considering a variant of the CHS model where the adversary does not get access to many copies of one Haar state but instead gets access to infinitely many input-less oracles[1]

[1] We note that [Kre21] made similar use of infinitely many oracles to prove a separation between pseudorandom states and one-way functions.

$\left\{\{G_{k,x}\}_{k,x\in\{0,1\}^\lambda}\right\}_{\lambda\in\mathbb{N}}$ such that each $G_{k,x}$ produces a copy of a Haar state $|\psi_{k,x}\rangle$. In this model, it is easy to construct pseudorandom function-like states. However, an extension of Theorem 5 rules out the possibility of interactive quantum key agreement and quantum commitments with classical communication in this variant. Thus, we have the following.

Theorem 6. *There does not exist a black-box reduction from interactive quantum key agreement and quantum commitments with classical communication to pseudorandom function-like states.*

Prior work by Chung, Goldin and Gray [CGG24] extensively studies the separations between quantum cryptographic primitives with classical communication and different quantum pseudorandomness notions. However, their framework did not capture the above result.

Prior works by [ACC+22, CLM23, LLLL24] ruled out quantum key agreements and non-interactive commitments with classical communication from post-quantum one-way functions. However, their separation was either based on a conjecture or in a restricted setting whereas our result is unconditional. This makes our result incomparable with the results from [ACC+22, CLM23, LLLL24]. Our work follows a long line of recent works [HY20, ACC+22, AHY23, CLM23, ACH+23, BGVV+23, BM+24, CM24] that make progress in understanding the landscape of black-box separations in quantum cryptography.

2 Technical Overview

2.1 Pseudorandomness in the CHS Model

Warmup: Pseudorandom State Generators (PRSGs). As a warmup, we first study 1-copy PRSG in the CHS model. Consider the following construction: $G_k(|\vartheta\rangle) := (Z^k \otimes I_{n-\lambda})|\vartheta\rangle$, where $Z^k = Z^{k_1} \otimes \cdots \otimes Z^{k_\lambda}$, $k = k_1 \cdots k_\lambda \in \{0,1\}^\lambda$ and $I_{n-\lambda}$ is an identity operator on $n-\lambda$ qubits. In other words, G_k applies a random Pauli Z operator only on the first λ qubits and does not touch the rest. Note that this construction already satisfies the stretch property (i.e. the output length is larger than the key length).

Let us consider the case when the adversary receives just one copy of $|\vartheta\rangle$ and is expected to distinguish $G_k(|\vartheta\rangle)$ versus an independent Haar state $|\varphi\rangle$. Formally, we would like to argue that the following states are close.

$$\rho := \mathop{\mathbb{E}}_{\substack{k\leftarrow\{0,1\}^\lambda \\ |\vartheta\rangle\leftarrow\mathcal{H}_n}} [G_k(|\vartheta\rangle) \otimes |\vartheta\rangle\langle\vartheta|] \text{ and } \sigma := \frac{I}{2^n} \otimes \frac{I}{2^n}.$$

By the properties of the symmetric subspace, the following holds:

$$\mathop{\mathbb{E}}_{|\vartheta\rangle\leftarrow\mathcal{H}_n} \left[|\vartheta\rangle\langle\vartheta|^{\otimes 2}\right] \approx_\varepsilon \mathop{\mathbb{E}}_{x,y\leftarrow[2^n],x^1\neq y^1} \left[\frac{1}{2}\left(|xy\rangle\langle xy| + |xy\rangle\langle yx| + |yx\rangle\langle xy| + |yx\rangle\langle yx|\right)\right],$$

where ε is negligible in n and the notation x^1 (respectively, y^1) denotes the first λ bits of x (respectively, y). Now, applying a random Z operator on the first λ qubits tantamounts to measuring the first λ qubits in the computational basis. Given the fact that $x^1 \neq y^1$, this measurement unentangles the last n qubits. Thus, the result is a state of the form $\mathbb{E}_{x,y\leftarrow[2^n],x^1\neq y^1}\left[\frac{1}{2}|x\rangle\langle x| \otimes |y\rangle\langle y| + \frac{1}{2}|y\rangle\langle y| \otimes |x\rangle\langle x|\right]$. This state is in turn close to $\frac{I}{2^n} \otimes \frac{I}{2^n}$.

Generalizing to Many Copies of the CHS. Next, we to generalize the above approach to even when polynomially many copies of the CHS are provided. Formally, we would like to argue that the following two states are close.

$$\rho := \mathop{\mathbb{E}}_{\substack{k\leftarrow\{0,1\}^\lambda \\ |\vartheta\rangle\leftarrow\mathcal{H}_n}} \left[G_k(|\vartheta\rangle) \otimes |\vartheta\rangle\langle\vartheta|^{\otimes t}\right] \text{ and } \sigma := \mathop{\mathbb{E}}_{\substack{|\varphi\rangle\leftarrow\mathcal{H}_n \\ |\vartheta\rangle\leftarrow\mathcal{H}_n}} \left[|\varphi\rangle\langle\varphi| \otimes |\vartheta\rangle\langle\vartheta|^{\otimes t}\right],$$

where t is some polynomial of n. Note that, by the property of the Haar distribution, we can simplify σ to

$$\sigma = \frac{I}{2^n} \otimes \mathop{\mathbb{E}}_{T\leftarrow[0:t]^N} |T\rangle\langle T|,$$

where $|T\rangle$ is a type state[2] and $N = 2^n$. Note that by the properties of the symmetric subspace,

$$\mathop{\mathbb{E}}_{|\vartheta\rangle\leftarrow\mathcal{H}_n} \left[|\vartheta\rangle\langle\vartheta|^{\otimes t+1}\right] \approx_\varepsilon \mathop{\mathbb{E}}_{\substack{T\leftarrow[0:t+1]^N \\ T \text{ is } \lambda\text{-prefix collision-free}}} |T\rangle\langle T|,$$

where ε is negligible in n and T is λ-prefix collision-free if $T \in \{0,1\}^N$ and for any $x, y \in T$[3] with $x \neq y$ implies $x^1 \neq y^1$, where the notation x^1 (respectively, y^1) denotes the first λ bits of x (respectively, y). Note that, any λ-prefix collision-free type T,

$$|T\rangle = \frac{1}{\sqrt{\binom{t+1}{t}}} \sum_{x\in T} |x\rangle|T \setminus \{x\}\rangle.$$

Again, applying a random Z operator on the first λ qubits tantamounts to measuring the first λ qubits in the computational basis. Given the fact that T is λ-prefix collision-free, this measurement unentangles the first n qubits. Thus, the result is a state of the form

$$\mathop{\mathbb{E}}_{\substack{T\leftarrow[0:t+1]^N \\ T \text{ is } \lambda\text{-prefix collision-free} \\ x\leftarrow T}} \left[|x\rangle\langle x| \otimes |T \setminus \{x\}\rangle\langle T \setminus \{x\}|\right].$$

This state is in turn close to $\frac{I}{2^n} \otimes \mathbb{E}_{T\leftarrow[0:t]^N} |T\rangle\langle T|$.

[2] We encourage readers unfamiliar with type states to refer to Definition 7.

[3] Since $T \in \{0,1\}^N$, we can treat it as a set, in particular the set associated to T is $\{i : T[i] = 1\}$.

Generalizing to ℓ-Copy PRSG. Finally, we generalize this ℓ-copy PRSG. Formally, we would like to argue that the following two states are close.

$$\rho := \mathop{\mathbb{E}}_{\substack{k\leftarrow\{0,1\}^\lambda \\ |\vartheta\rangle\leftarrow\mathcal{H}_n}} \left[G_k(|\vartheta\rangle)^{\otimes\ell} \otimes |\vartheta\rangle\langle\vartheta|^{\otimes t}\right] \text{ and } \sigma := \mathop{\mathbb{E}}_{\substack{|\varphi\rangle\leftarrow\mathcal{H}_n \\ |\vartheta\rangle\leftarrow\mathcal{H}_n}} \left[|\varphi\rangle\langle\varphi|^{\otimes\ell} \otimes |\vartheta\rangle\langle\vartheta|^{\otimes t}\right],$$

where ℓ, t is some polynomial of n. Note that, by the property of the Haar distribution, we can simplify σ to

$$\sigma = \mathop{\mathbb{E}}_{T_1\leftarrow[0:\ell]^N} |T_1\rangle\langle T_1| \otimes \mathop{\mathbb{E}}_{T_2\leftarrow[0:t]^N} |T_2\rangle\langle T_2|,$$

where $|T_1\rangle, |T_2\rangle$ are type states and $N = 2^n$. Note that, similar to the last case, we can still write,

$$\mathop{\mathbb{E}}_{|\vartheta\rangle\leftarrow\mathcal{H}_n} \left[|\vartheta\rangle\langle\vartheta|^{\otimes t+\ell}\right] \approx_\varepsilon \mathop{\mathbb{E}}_{\substack{T\leftarrow[0:t+\ell]^N \\ T \text{ is } \lambda\text{-prefix collision-free}}} |T\rangle\langle T|,$$

and any λ-prefix collision-free type T,

$$|T\rangle = \frac{1}{\sqrt{\binom{t+\ell}{\ell}}} \sum_{\substack{T_1\subset T \\ |T_1|=\ell}} |T_1\rangle|T \setminus T_1\rangle.$$

Ideally, we would want the application of $(Z^k \otimes I_{n-\lambda})^{\otimes\ell}$ to unentangle $|T_1\rangle$ from $|T \setminus T_1\rangle$. This is equivalent to measuring the first ℓ registers in the type basis. This is in general not true, not true. Hence, we settle for the next best thing, which is finding a "dense-enough"[4] subset of λ-prefix collision-free type such that $(Z^k \otimes I_{n-\lambda})^{\otimes\ell}$ to unentangle $|T_1\rangle$ from $|T \setminus T_1\rangle$. We find this subset to be "λ-prefix ℓ-fold collision-free" types.

We say that a λ-prefix collision-free type T is "λ-prefix ℓ-fold collision-free" if for all pairs of ℓ sized subsets $T_1, T_2 \subset T$, $\oplus_{x\in T_1} x = \oplus_{x\in T_2} x$ only if $T_1 = T_2$. We start by noting that this subset is only "dense-enough" if $\ell = O\left(\frac{\lambda}{\log(\lambda)^{1+\varepsilon}}\right)$, for any constant $\varepsilon > 0$.[5]

Next, we show that for these λ-prefix ℓ-fold collision-free types states, applying a random $(Z^k \otimes I_{n-\lambda})^{\otimes\ell}$ is equivalent to measuring the first ℓ registers in the type basis. This is because $(Z^k \otimes I_{n-\lambda})^{\otimes\ell}$ on a type state $|T_1\rangle$ is equivalent to adding a phase of $(-1)^{k\cdot(\oplus_{x\in T_1} x)}$. Hence,

$$\mathop{\mathbb{E}}_k \left[(Z^k \otimes I_{n-\lambda})^{\otimes\ell} \otimes I_{tn} |T\rangle\langle T| (Z^k \otimes I_{n-\lambda})^{\otimes\ell} \otimes I_{tn}\right]$$

$$= \mathop{\mathbb{E}}_k \left[\frac{1}{\binom{t+\ell}{\ell}} \sum_{\substack{T_1,T_2\subset T \\ |T_1|=|T_2|=\ell}} (-1)^{k\cdot(\oplus_{x\in T_1} x \oplus \oplus_{y\in T_2} y)} |T_1\rangle|T \setminus T_1\rangle\langle T_2|\langle T \setminus T_2|\right],$$

[4] Here, by dense-enough, we mean when picking a random type from λ-prefix collision-free, it lies in this subset with probability $1 - \mathsf{negl}$.

[5] Later, in the impossibility result, we show that this is in fact the best we can hope for as a larger subset would bypass the impossibility result.

which for λ-prefix ℓ-fold collision-free types states is non-zero only if $T_1 = T_2$, giving us

$$\mathop{\mathbb{E}}_{k}\left[(Z^k \otimes I_{n-\lambda})^{\otimes \ell} \otimes I_{tn} |T\rangle\langle T| (Z^k \otimes I_{n-\lambda})^{\otimes \ell} \otimes I_{tn}\right] = \mathop{\mathbb{E}}_{\substack{T_1 \subset T \\ |T_1|=\ell}} \left[|T_1\rangle\langle T_1| \otimes |T \setminus T_1\rangle\langle T \setminus T_1|\right].$$

Over expectation over all λ-prefix ℓ-fold collision-free types states, this state is close to $\mathbb{E}_{T_1 \leftarrow [0:\ell]^N} |T_1\rangle\langle T_1| \otimes \mathbb{E}_{T_2 \leftarrow [0:t]^N} |T_2\rangle\langle T_2|$.

Limitations. To complement our result, we show that a t-copy PRSG is impossible in the CHS model, for $\ell = O\left(\frac{\lambda}{\log(\lambda)}\right)$ (for a restricted class of PRSG constructs which only takes one copy of the common Haar state). We show this by showing that the rank of σ grows much faster than the rank of ρ, hence, a simple distinguisher is a projector on the eigenspace of ρ. In particular, let $\tilde{G}_k(\vartheta)$ be the PRSG. Then define

$$\rho := \mathop{\mathbb{E}}_{\substack{k \leftarrow \{0,1\}^\lambda \\ |\vartheta\rangle \leftarrow \mathcal{H}_n}} \left[\tilde{G}_k(|\vartheta\rangle)^{\otimes \ell} \otimes |\vartheta\rangle\langle\vartheta|^{\otimes t}\right] \text{ and } \sigma := \mathop{\mathbb{E}}_{\substack{|\varphi\rangle \leftarrow \mathcal{H}_n \\ |\vartheta\rangle \leftarrow \mathcal{H}_n}} \left[|\varphi\rangle\langle\varphi|^{\otimes \ell} \otimes |\vartheta\rangle\langle\vartheta|^{\otimes t}\right]$$

Now since $\tilde{G}_k(|\vartheta\rangle)$ is a PRSG, its output is negligibly close to a pure state. This means that the rank of $\rho \leq 2^\lambda \binom{2^n+t+\ell-1}{t+\ell}$. In contrast, the rank of $\sigma = \binom{2^n+\ell-1}{\ell}\binom{2^n+t-1}{t}$. Note that, for $t = \lambda^3$ and $\ell = \lambda/\log(\lambda)$, $\mathrm{rank}(\rho)/\mathrm{rank}(\sigma) = \mathsf{negl}$. Hence, we can find a distinguisher. Here the distinguisher just projects onto the eigenspace of ρ, ρ gets accepted with probability 1 but σ gets accepted with probability negl, hence giving a disguisher. Since PRFSs imply PRSs (by setting $c = 0$), achieving an ℓ-query statistical PRFS in the CHS model for $\ell = \Omega(\lambda/\log(\lambda))$ is impossible.

Pseudorandom Function-like State Generators. Next we extend this idea from PRSGs to achieve PRFSGs. We take inspiration from the seminal Goldreich-Goldwasser-Micali approach [GGM86]. In particular, on the key $K = (k_1^0, \ldots, k_m^0, k_1^1, \ldots, k_m^1) \in \{0,1\}^{2\lambda' m}$ and the input $\mathbf{x} = (x_1, \ldots, x_m) \in \{0,1\}^m$, define the PRFSG $G_K(\mathbf{x}, |\vartheta\rangle)$ as follows: $G_K(\mathbf{x}, |\vartheta\rangle) = (Z^{\bigoplus_{i=1}^m k_i^{x_i}} \otimes I_{n-\lambda'})|\vartheta\rangle$. Formally, the following two states are close:

$$\rho := \mathop{\mathbb{E}}_{\substack{K \leftarrow \{0,1\}^{2\,m\lambda'} \\ |\vartheta\rangle \leftarrow \mathcal{H}_n}} \left[\otimes_{i=1}^q G_K(\mathbf{x}^i, |\vartheta\rangle)^{\otimes \ell_i} \otimes |\vartheta\rangle\langle\vartheta|^{\otimes t}\right],$$

and

$$\sigma := \mathop{\mathbb{E}}_{\substack{\forall i \in [q], |\varphi_i\rangle \leftarrow \mathcal{H}_n \\ |\vartheta\rangle \leftarrow \mathcal{H}_n}} \left[\otimes_{i=1}^q |\varphi_i\rangle\langle\varphi_i|^{\otimes \ell_i} \otimes |\vartheta\rangle\langle\vartheta|^{\otimes t}\right],$$

for all $\mathbf{x}^1, \ldots, \mathbf{x}^q \in \{0,1\}^m$ and $\ell_1, \ldots, \ell_q$ such that $\sum_{i=1}^q \ell_i = \ell$, for $\ell = O\left(\frac{\lambda^{1-c}}{\log(\lambda)^{1+\varepsilon}}\right)$ and $m = \lambda^c$, for any constant $\varepsilon > 0$ and $c \in [0,1)$.

Just as before, we can write σ as follows:

$$\sigma = \bigotimes_{i=1}^{q} \mathop{\mathbb{E}}_{T_i \leftarrow [0:\ell_i]^N} |T_i\rangle\langle T_i| \otimes \mathop{\mathbb{E}}_{\tilde{T} \leftarrow [0:t]^N} |\tilde{T}\rangle\langle \tilde{T}|,$$

where T_i's and $\tilde{T}$ are type states and $N = 2^n$. Note that, similar to the last case, we can still write,

$$\mathop{\mathbb{E}}_{|\vartheta\rangle \leftarrow \mathcal{H}_n} \left[|\vartheta\rangle\langle\vartheta|^{\otimes t+\ell} \right] \approx_\varepsilon \mathop{\mathbb{E}}_{\substack{T \leftarrow [0:t+\ell]^N \\ T \text{ is } \lambda\text{-prefix } \ell\text{-fold collision-free}}} |T\rangle\langle T|,$$

and any λ-prefix ℓ-fold collision-free type T,

$$|T\rangle = \frac{1}{\sqrt{\binom{t+\ell}{\ell}}} \sum_{\substack{T_1 \subset T \\ |T_1| = \ell}} |T_1\rangle |T \setminus T_1\rangle.$$

Now, after application of one layer of $(Z^k \otimes I_{n-\lambda})^{\otimes \ell}$, we know that $|T_1\rangle$ unentangles from $|T \setminus T_1\rangle$. We extend this idea to show that even for a tensor of type states, applying $(Z^k \otimes I_{n-\lambda})^{\otimes \tilde{\ell}_i}$ on parts of each type state still unentangles each of them as long as all the type states are λ-prefix ℓ-fold collision-free type and their combined set is still λ-prefix ℓ-fold collision-free. Formally, we show the following: Let $\tilde{\ell}_1, \ldots, \tilde{\ell}_q \in \mathbb{N}$, and $t_1, \ldots, t_q \in \mathbb{N}$ such that $\sum_{i=1}^q \tilde{\ell}_i = \tilde{\ell}$ and $\sum_{i=1}^q t_i = t$. Then for any λ-prefix $\tilde{\ell}$-fold collision-free type T and any mutually disjoint sets $T_1, \ldots, T_q$ satisfying $\bigcup_{i=1}^q T_i = T$ and $|T_i| = t_i + \tilde{\ell}_i$ for all $i \in [q]$,

$$\mathop{\mathbb{E}}_{k \leftarrow \{0,1\}^n} \left[\bigotimes_{i=1}^{q} \left(\left(Z^k \otimes I_m\right)^{\otimes \ell_i} \otimes I_{n+m}^{\otimes t_i} \right) |T_i\rangle\langle T_i| \left(\left(Z^k \otimes I_m\right)^{\otimes \tilde{\ell}_i} \otimes I_{n+m}^{\otimes t_i} \right) \right]$$
$$= \bigotimes_{i=1}^{q} \mathop{\mathbb{E}}_{\substack{X_i \subset T_i \\ |X_i| = \tilde{\ell}_i}} \left[|X_i\rangle\langle X_i| \otimes |T_i \setminus X_i\rangle\langle T_i \setminus X_i| \right].$$

Hence, applying each layer $(Z^{k_i^b} \otimes I_{n-\lambda})$ unentangles all type states into two halfs. Hence, by repeated application, we get

$$\rho \approx_\varepsilon \mathop{\mathbb{E}}_{\substack{T \leftarrow [0:t+\ell]^N \\ T \text{ is } \lambda\text{-prefix } \ell\text{-fold collision-free}}} \mathop{\mathbb{E}}_{(T_1, T_2, \ldots, T_q, \hat{T})} \left[\bigotimes_{i=1}^{q} |T_i\rangle\langle T_i| \otimes |\hat{T}\rangle\langle \hat{T}| \right],$$

where $(T_1, T_2, \ldots, T_q, \hat{T})$ are sampled as follows: for $i = 1, 2, \ldots, q$, sample an ℓ_i-subset from $T \setminus (\bigcup_{j=1}^{i-1} T_j)$ uniformly and let $\hat{T} := T \setminus (\bigcup_{j=1}^{q} T_j)$. Over expectation over all λ-prefix ℓ-fold collision-free types states, this state is close to σ.

2.2 Quantum Bit Commitments

With t-copy PRSG in hand, we construct a statistically-hiding, statistically-binding commitment scheme in the CHS model. Our scheme draws inspiration from the quantum commitment scheme introduced in [MY21, MNY23] that builds quantum bit commitments from t-copy PRSG.

In particular, to commit to $b = 0$, the committer creates a superposition over all keys of the PRSG in the decommitment register and runs the PRSG in superposition over this register. The committer sets this as the commitment register. To commit to $b = 1$, the committer creates a maximally entangled state over the commitment and the decommitment register. Formally,

$$|\psi_0\rangle_{\mathsf{C}_i\mathsf{R}_i} := \frac{1}{\sqrt{2^\lambda}} \sum_{k\in\{0,1\}^\lambda} G_k(|\vartheta\rangle)_{\mathsf{C}_i}|k||0^{n-\lambda}\rangle_{\mathsf{R}_i}$$

and

$$|\psi_1\rangle_{\mathsf{C}_i\mathsf{R}_i} := \frac{1}{\sqrt{2^n}} \sum_{j\in\{0,1\}^n} |j\rangle_{\mathsf{C}_i}|j\rangle_{\mathsf{R}_i},$$

where, $(\mathsf{C}_1, \ldots, \mathsf{C}_p)$ is the commitment register and $(\mathsf{R}_1, \ldots, \mathsf{R}_p)$ is the reveal register.

To achieve hiding, our scheme relies on the pseudorandomness property of the PRSG. In particular, the commitment is very close to one where the keys are distinct for all $(\mathsf{C}_i, \mathsf{R}_i)$, in this case, one copy of PRS is indistinguishable from a maximally mixed state.[6]

Unlike the approach in [MY21], our construction is not of the canonical form [Yan2]. To achieve binding, the receiver performs multiple SWAP tests. In particular, we show that since the rank of the commitment registers is exponentially separated, multiple SWAP tests can distinguish between the two.

2.3 Black-Box Separations

LOCC Indistinguishability. The notion of LOCC indistinguishability is well-studied and is referred to as quantum data hiding by quantum information theorists [BDF+99, DLT02, EW02, Gea02, HLS05, MWW09, CLMO13, PNC14, CH14, CLM+14, HBAB19]. In this setting, there is a challenger, two (possibly entangled and mixed) bipartite quantum states ρ_{AB} and σ_{AB}, and a computationally unbounded, two-party distinguisher (Alice, Bob) who are spatially separated and without pre-shared entanglement. The challenger picks a quantum state from $\{\rho_{\mathsf{AB}}, \sigma_{\mathsf{AB}}\}$ uniformly at random and sends register A to Alice and register B to Bob respectively. The task of Alice and Bob is to distinguish whether they are given ρ_{AB} or σ_{AB} by performing local operations and communicating classically. We call such distinguishers *LOCC adversaries.*

We focus on the case where Alice and Bob each receive $t = \mathrm{poly}(\lambda)$ copies of $|\psi\rangle_{\mathsf{A}}$ and $|\phi\rangle_{\mathsf{B}}$, where $|\psi\rangle$ and $|\phi\rangle$ are either two identical or i.i.d. Haar states of length $n = \omega(\log(\lambda))$. Explicitly, the two input states are

$$\rho_{\mathsf{AB}} = \mathop{\mathbb{E}}_{|\psi\rangle\leftarrow\mathcal{H}_n} \left[|\psi\rangle\langle\psi|_{\mathsf{A}}^{\otimes t} \otimes |\psi\rangle\langle\psi|_{\mathsf{B}}^{\otimes t}\right],$$

[6] Note that this still needs multi-key security which is not trivial in the CHS model, since all the PRS generators share the same Haar state for randomness. But we prove that our construction satisfies multikey security.

$$\sigma_{\mathsf{AB}} = \mathop{\mathbb{E}}_{|\psi\rangle\leftarrow\mathcal{H}_n}\left[|\psi\rangle\langle\psi|_{\mathsf{A}}^{\otimes t}\right] \otimes \mathop{\mathbb{E}}_{|\phi\rangle\leftarrow\mathcal{H}_n}\left[|\phi\rangle\langle\phi|_{\mathsf{B}}^{\otimes t}\right].$$

Note that if global measurements are allowed, performing SWAP tests can easily distinguish them. As one of our main technical contributions, we show that for any LOCC adversary, the advantage of distinguishing ρ_{AB} from σ_{AB} is negligible in λ. Before we explain the proof, we compare our theorem with [Har23, Theorem 8]. In short, the theorems are incomparable. Our setting is stronger in the sense that the LOCC adversary both obtain polynomial copies of the input, while [Har23, Theorem 8] studies the single-copy setting. However, [Har23, Theorem 8] is more general since it holds for a family of input states, whereas the input in our setting is fixed to ρ_{AB} and σ_{AB}, which are belong to the family.

Toward the proof, we start by using the following common technique in proving LOCC indistinguishability: the set of LOCC measurements is a (proper) subset of the set of all positive partial transpose (PPT) measurements [CLM+14]. Hence, it is sufficient to upper bound the maximum distinguishing advantage over two-outcome PPT measurements, i.e., $\{M_{\mathsf{AB}}, I_{\mathsf{AB}} - M_{\mathsf{AB}}\}$ such that $0 \preceq M_{\mathsf{AB}} \preceq I_{\mathsf{AB}}$ and $0 \preceq M_{\mathsf{AB}}^{\Gamma_B} \preceq I_{\mathsf{AB}}$, where $M_{\mathsf{AB}}^{\Gamma_B}$ denote the partial transpose of M_{AB} with respect to B. Next, from the basic properties of partial transpose and trace norm, we show that the distinguishing advantage is bounded by the trace norm between $\rho_{\mathsf{AB}}^{\Gamma_{\mathsf{B}}}$ and $\sigma_{\mathsf{AB}}^{\Gamma_{\mathsf{B}}}$.

The most technical part of the proof is to upper bound the quantity $\left\|\rho_{\mathsf{AB}}^{\Gamma_{\mathsf{B}}} - \sigma_{\mathsf{AB}}^{\Gamma_{\mathsf{B}}}\right\|_1$. We point out that the partial transpose of a density matrix might *not* be a positive semidefinite matrix. Our first step is to expand ρ_{AB} and σ_{AB} in the *type basis* as follows:

$$\rho_{\mathsf{AB}} = \mathop{\mathbb{E}}_{T\leftarrow[0:2t]^d}\left[|T\rangle\langle T|_{\mathsf{AB}}\right],$$

$$\sigma_{\mathsf{AB}} = \mathop{\mathbb{E}}_{S_A\leftarrow[0:t]^d}\left[|S_A\rangle\langle S_A|_{\mathsf{A}}\right] \otimes \mathop{\mathbb{E}}_{S_B\leftarrow[0:t]^d}\left[|S_B\rangle\langle S_B|_{\mathsf{B}}\right],$$

where $d := 2^n$. Next, we further conditioned on the events that (1) T, S_A and S_B each have no repeated elements (2) S_A and S_B have no identical elements. From the collision bound, doing so only incurs an additional error of $O(t^2/d) = \mathsf{negl}(\lambda)$. Therefore, we can now treat T, S_A and S_B as *sets*. It suffices to prove that $\left\|\tilde{\rho}_{\mathsf{AB}}^{\Gamma_{\mathsf{B}}} - \tilde{\sigma}_{\mathsf{AB}}^{\Gamma_{\mathsf{B}}}\right\|_1$ is negligible in λ, where

$$\tilde{\rho}_{\mathsf{AB}} := \mathop{\mathbb{E}}_{T\leftarrow\binom{[d]}{2t}}\left[|T\rangle\langle T|_{\mathsf{AB}}\right],$$

$$\tilde{\sigma}_{\mathsf{AB}} := \mathop{\mathbb{E}}_{\substack{S_A,S_B\leftarrow\binom{[d]}{t}:\\ S_A\cap S_B=\emptyset}}\left[|S_A\rangle\langle S_A|_{\mathsf{A}} \otimes |S_B\rangle\langle S_B|_{\mathsf{B}}\right].$$

Observe that the $\tilde{\sigma}_{\mathsf{AB}}^{\Gamma_B} = \tilde{\sigma}_{\mathsf{AB}}$. To obtain a simpler expression of $\tilde{\rho}_{\mathsf{AB}}^{\Gamma_B}$, we rely on the following useful identity for bi-partitioning the type states:

$$|T\rangle_{\mathsf{AB}} = \sum_{X\in\binom{T}{t}} \frac{1}{\sqrt{\binom{2t}{t}}} |T\setminus X\rangle_{\mathsf{A}} \otimes |X\rangle_{\mathsf{B}}.$$

Hence, the partial transpose of $\tilde{\rho}_{\mathsf{AB}}$ can be written as

$$\tilde{\rho}_{\mathsf{AB}}^{\Gamma_B} = \mathop{\mathbb{E}}_{T \leftarrow \binom{[d]}{2t}} \left[\frac{1}{\binom{2t}{t}} \sum_{X,Y \in \binom{T}{t}} |T \setminus X\rangle\langle T \setminus Y|_{\mathsf{A}} \otimes |Y\rangle\langle X|_{\mathsf{B}} \right].$$

If $X = Y$, then the term is the tensor product of two *disjoint* sets $|T \setminus X\rangle\langle T \setminus X|_{\mathsf{A}} \otimes |X\rangle\langle X|_{\mathsf{B}}$. Such a term will be canceled out by the corresponding term in $\tilde{\sigma}_{\mathsf{AB}}^{\Gamma_B}$ since they have equal coefficients. Therefore, the difference between them is the following matrix with mismatched X and Y:

$$\tilde{\rho}_{\mathsf{AB}}^{\Gamma_B} - \tilde{\sigma}_{\mathsf{AB}}^{\Gamma_B} = \mathop{\mathbb{E}}_{T \leftarrow \binom{[d]}{2t}} \left[\frac{1}{\binom{2t}{t}} \sum_{X,Y \in \binom{T}{t} : X \neq Y} |T \setminus X\rangle\langle T \setminus Y|_{\mathsf{A}} \otimes |Y\rangle\langle X|_{\mathsf{B}} \right].$$

We continue to simplify it by applying a double-counting argument. Every tuple of sets (T, X, Y) uniquely determines a tuple of mutually disjoint sets (C, I, X', Y') satisfying $C = T \setminus (X \cup Y)$ (C for the complement of $X \cup Y$), $I = X \cap Y$ (I for intersection), $X' = X \setminus I$ and $Y' = Y \setminus I$. Hence, $T \setminus X = C \uplus Y'$, $Y = I \uplus Y'$, $T \setminus Y = C \uplus X'$, and $X = I \uplus X'$ where $\uplus$ denotes the disjoint union. By further classifying the summands according to $s := |C| = |I| \in \{0, 1, \ldots, t-1\}$ (note that then $|X'| = |Y'| = t - s$), we have

$$\begin{aligned}
&\left\| \tilde{\rho}_{\mathsf{AB}}^{\Gamma_{\mathsf{B}}} - \tilde{\sigma}_{\mathsf{AB}}^{\Gamma_{\mathsf{B}}} \right\|_1 \\
&= \frac{1}{\binom{d}{2t}\binom{2t}{t}} \left\| \sum_{s=0}^{t-1} \sum_{C \in \binom{[d]}{s}} \sum_{I \in \binom{[d] \setminus C}{s}} \sum_{\substack{X', Y' \in \binom{[d] \setminus (C \uplus I)}{t-s}: \\ X' \cap Y' = \emptyset}} |C \uplus Y'\rangle_{\mathsf{A}} |I \uplus Y'\rangle_{\mathsf{B}} \langle C \uplus X'|_{\mathsf{A}} \langle I \uplus X'|_{\mathsf{B}} \right\|_1 \\
&\leq \frac{1}{\binom{d}{2t}\binom{2t}{t}} \sum_{s=0}^{t-1} \sum_{C \in \binom{[d]}{s}} \sum_{I \in \binom{[d] \setminus C}{s}} \underbrace{\left\| \sum_{\substack{X', Y' \in \binom{[d] \setminus (C \uplus I)}{t-s}: \\ X' \cap Y' = \emptyset}} |C \uplus Y'\rangle_{\mathsf{A}} |I \uplus Y'\rangle_{\mathsf{B}} \langle C \uplus X'|_{\mathsf{A}} \langle I \uplus X'|_{\mathsf{B}} \right\|_1}_{=: K_{C,I}},
\end{aligned}$$

where the inequality follows from the triangle inequality. We observe that the matrix $K_{C,I}$ has the same structure as the adjacency matrix of *Kneser graphs*. Here, we recall the definition of Kneser graphs. For $v, k \in \mathbb{N}$, the Kneser graph $K(v, k)$ is the graph whose vertices correspond to the k-element subsets of the set $[v]$, and two vertices are adjacent if and only if the two corresponding sets are disjoint. Therefore, for every (C, I), the matrix $K_{C,I}$ is isospectral to the adjacency matrix of the Kneser graph $K(d - |C| - |I|, t - |I|)$. Finally, we employ the well-studied spectral property of Kneser graphs as a black box to obtain an $O(t^2/d) = \mathsf{negl}(\lambda)$ upper bound for $\left\| \tilde{\rho}_{\mathsf{AB}}^{\Gamma_{\mathsf{B}}} - \tilde{\sigma}_{\mathsf{AB}}^{\Gamma_{\mathsf{B}}} \right\|_1$.

Furthermore, we show the tightness of the theorem by constructing an optimal LOCC distinguisher that achieves the same advantage. The strategy is simple: Alice and Bob each individually measure every copy of their input in the

computational basis and obtain a total of $2t$ outcomes. Then, they output 1 if there is any collision among these $2t$ outcomes.

Impossibility Results in the CHS Model. With the LOCC Haar indistinguishability theorem in hand, we investigate the limits of the CHS model when the communication between the parties is classical. We show that the several impossibility results of information-theoretically secure schemes in the plain model can be generically lifted to the CHS model, even when the adversary does not receive any common Haar state. We emphasize that there is no classical counterpart in the CRS model. If the adversary is not given the CRS, then many information-theoretically secure schemes do exist, such as key agreements.

As common in proving impossibilities, our approach is to convert schemes in the CHS model to those in the plain model. The transform is simple: in the new scheme, the parties each sample polynomially many copies of the Haar state *independently* and run the original scheme. Crucially, despite the inconsistency in their Haar states, the new scheme still satisfies completeness thanks to the LOCC Haar indistinguishability. A caveat is that sampling Haar states is time-inefficient. However, since the impossibilities in the plain model are still valid if the (honest) algorithms in the scheme are time-inefficient, doing so is acceptable for the sake of showing impossibilities.

Separation Results. We separate many important primitives from $(\lambda, \omega(\log(\lambda)))$-PRSG. Since $(\lambda, \omega(\log(\lambda)))$-PRSGs do not exist in the CHS model, we need to "strengthen" the oracle in order to prove separations. For every security parameter $\lambda \in \mathbb{N}$, we define the oracle as $\{G_k\}_{k\in\{0,1\}^\lambda}$ where each G_k is an isometry that takes no input and outputs an i.i.d. Haar state $|\psi_k\rangle$.

Relative to this oracle, the implementation of the PRSG is straightforward: the output on k of any length $\lambda \in \mathbb{N}$ is $|\psi_k\rangle$. The security directly follows from the hardness of unstructured search. To prove the non-existence of QCCC schemes, we employ a two step approach. First, showing that a scheme with respect to this oracle can be transformed to schemes with respect to a much weaker oracle. Second, showing that this much weaker oracle does not give much extra power over the plain model. Formally: First, similar to the previous section, we show that due to the LOCC indistinguishability, the parties can sample all "large" quantum states on their own, and the correctness and security is only "polynomially" affected[7]. This means that any scheme with respect this oracle can be turned into a scheme with respect to an oracle with only short (constant times logarithmic) Haar states. Second, for short (constant times logarithmic) quantum states, we show that this oracle does not give much extra power since an adversary can learn the oracle completely. This is because for short-enough states, the adversary can run tomography on polynomial queries and learn the state with up to inverse polynomial error. Hence, the adversary can simulate

[7] Since the Haar indistinguishability has a factor of $O(t^2/d)$, as long as t^2/d is inverse-polynomial, we do not incur a lot of loss.

both parties post-selecting on a transcript to learn any secret[8]. This means that any scheme secure in the presence of this oracle can be transformed into another scheme that is secure in the plain model.

Lastly, we observe that by considering a generalized oracle, namely $\{G_{k,x}\}_{k,x\in\{0,1\}^\lambda}$, we can show that (classically accessible) PRFSGs with super-logarithmic input length exist. We can extend the impossibility of QCCC commitments to hold in the presence of the generalized oracle as well. Thus, we can separate PRFS and QCCC commitments.

3 Preliminaries

We denote the security parameter by λ. We assume that the reader is familiar with the fundamentals of quantum computing covered in [NC10].

3.1 Notation

- We use $[n]$ to denote $\{1,\ldots,n\}$ and $[0:n]$ to denote $\{0,1,\ldots,n\}$.
- For any finite set T and any integer $0 \leq k \leq |T|$, we denote by $\binom{T}{k}$ the set of all k-size subsets of T.
- For any finite set T, we use the notation $x \leftarrow T$ to indicate that x is sampled uniformly from T.
- We denote by S_t the symmetric group of degree t.
- For any set A and $t \in \mathbb{N}$, we denote by A^t the t-fold Cartesian product of A.
- For $\sigma \in S_t$ and $\mathbf{v} = (v_1,\ldots,v_t)$, we define $\sigma(\mathbf{v}) := (v_{\sigma(1)},\ldots,v_{\sigma(t)})$.
- We denote by $\mathcal{D}(H)$ the set of density matrices in the Hilbert space H.
- Let $\rho_{AB} \in \mathcal{D}(H_A \otimes H_B)$, by $\mathrm{Tr}_B(\rho_{AB}) \in \mathcal{D}(H_A)$ we denote the reduced density matrix by taking partial trace over B.
- We denote by $\mathsf{TD}(\rho,\rho') := \frac{1}{2}\|\rho-\rho'\|_1$ the trace distance between quantum states ρ,ρ', where $\|X\|_1 = \mathrm{Tr}(\sqrt{X^\dagger X})$ denotes the trace norm.
- For any matrices A, B, we write $A \preceq B$ to indicate that $B - A$ is positive semi-definite.
- For any Hermitian matrix O, the trace norm of O has the following variational definition:
$$\|O\|_1 = \max_{-I\preceq M\preceq I} \mathrm{Tr}(MO).$$
Furthermore, if $\mathrm{Tr}(O) = 0$ then $\|O\|_1 = 2\cdot\max_{0\preceq M\preceq I}\mathrm{Tr}(MO)$.
- We denote the Haar measure over n qubits by $\mathcal{H}_n$.
- For any matrix $M_{\mathsf{AB}} = \sum_{i,j,k,\ell}\alpha_{ijk\ell}|i\rangle\langle j|_{\mathsf{A}}\otimes|k\rangle\langle\ell|_{\mathsf{B}}$ on registers (A,B), by $M_{\mathsf{AB}}^{\Gamma_{\mathsf{B}}}$ we denote its *partial transpose* with respect to register B, i.e., $M_{\mathsf{AB}}^{\Gamma_{\mathsf{B}}} = \sum_{i,j,k,\ell}\alpha_{ijk\ell}|i\rangle\langle j|_{\mathsf{A}}\otimes|\ell\rangle\langle k|_{\mathsf{B}}$.[9]

[8] Note that since the adversary does not need to be efficient, as long as they have the description of this oracle, they can post-select on the transcript.

[9] Note that the (partial) transpose operation needs to be defined with respect to to an orthogonal basis. Throughout this work, it is always defined with respect to to the computational basis.

3.2 Common Haar State Model

The Common Haar State (CHS) model is related to the Common Reference Quantum State (CRQS) model [MNY23]. In this model, all parties receive polynomially many copies of a *single* quantum state sampled from the Haar distribution. Recently, another work of Chen et al. [CCS24] studied a similar model called the Common Haar Random State (CHRS) model. In the CHRS model, every party receives polynomially many copies of *polynomially many* i.i.d. Haar states.

We define another variant of the CHS model called the *Keyed* Common Haar State Model. In this model, all parties (once the security parameter is set to λ) have access to the oracle (called the *Keyed* Common Haar State Oracle) $G^\lambda := \{G_k\}_{k\in\{0,1\}^\lambda}$ as follows. For every $k \in \{0,1\}^\lambda$, the oracle G_k is a Haar isometry that maps any state $|\psi\rangle$ to $|\psi\rangle|\vartheta_k\rangle$, where $|\vartheta_k\rangle$ is a Haar state of length $n(\lambda) = \omega(\log(\lambda))$.

While the above variant is harder to instantiate (hence not useful for constructions), is a natural candidate for black-box separations as seen is Sect. 9.

Pseudorandom State (PRS) Generators in the CHS Model

Definition 1 (Statistically Secure (λ, n, ℓ)-Pseudorandom State Generators in the CHS Model). *We say that a QPT algorithm G is a* statistically secure (λ, n, ℓ)-pseudorandom state generator (PRSG) *in the CHS model if the following holds:*

- ***State Generation**: For any $\lambda \in \mathbb{N}$ and $k \in \{0,1\}^\lambda$, the algorithm G_k (where G_k denotes $G(k,\cdot)$) is a quantum channel such that for every $n(\lambda)$-qubit state $|\vartheta\rangle$,*
$$G_k(|\vartheta\rangle\langle\vartheta|) = |\vartheta_k\rangle\langle\vartheta_k|,$$
for some $n(\lambda)$-qubit state $|\vartheta_k\rangle$. We sometimes write $G_k(|\vartheta\rangle)$ for brevity.[10]
- ***ℓ-copy Pseudorandomness**: For any polynomial $t(\cdot)$ and any non-uniform, unbounded adversary $A = \{A_\lambda\}_{\lambda\in\mathbb{N}}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that:*
$$\left| \Pr_{\substack{k\leftarrow\{0,1\}^\lambda \\ |\vartheta\rangle\leftarrow\mathcal{H}_{n(\lambda)}}} \left[A_\lambda\left(G_k(|\vartheta\rangle)^{\otimes\ell(\lambda)} \otimes |\vartheta\rangle\langle\vartheta|^{\otimes t(\lambda)}\right) = 1\right] - \Pr_{\substack{|\varphi\rangle\leftarrow\mathcal{H}_{n(\lambda)} \\ |\vartheta\rangle\leftarrow\mathcal{H}_{n(\lambda)}}} \left[A_\lambda\left(|\varphi\rangle\langle\varphi|^{\otimes\ell(\lambda)} \otimes |\vartheta\rangle\langle\vartheta|^{\otimes t(\lambda)}\right) = 1\right] \right| \leq \mathsf{negl}(\lambda).$$

[10] More generally, the generation algorithm could take multiple copies of the common Haar state as input or output a state of different size compared to the common Haar state. Here, we focus on a restricted class of generators that only require a single copy of the common Haar state as input, and the output of the generator matches the size of the common Haar states.

If G satisfies ℓ-copy pseudorandomness for every polynomial $\ell(\cdot)$ then we drop ℓ from the notation and simply denote it to be a (λ, n)-PRSG.

We define a stronger definition below called *multi-key ℓ-copy PRS generators*. Looking ahead, our construction of PRS in Sect. 4.2 satisfies this definition.

Definition 2 (Multi-key Statistically Secure (λ, n, ℓ)-Pseudorandom State Generators in the CHS Model). *We say that a QPT algorithm G is a* multi-key statistically secure (λ, n, ℓ)-pseudorandom state generator *in the CHS model if the following holds:*

- ***State Generation**: For any $\lambda \in \mathbb{N}$ and $k \in \{0,1\}^\lambda$, the algorithm G_k (where G_k denotes $G(k, \cdot)$) is a quantum channel such that for every $n(\lambda)$-qubit state $|\vartheta\rangle$,*

$$G_k(|\vartheta\rangle\langle\vartheta|) = |\vartheta_k\rangle\langle\vartheta_k|,$$

  for some $n(\lambda)$-qubit state $|\vartheta_k\rangle$. We sometimes write $G_k(|\vartheta\rangle)$ for brevity.
- ***Multi-key ℓ-copy Pseudorandomness**: For any polynomial $t(\cdot)$, $p(\cdot)$ and any non-uniform, unbounded adversary $A = \{A_\lambda\}_{\lambda\in\mathbb{N}}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that:*

$$\left| \Pr_{\substack{k_1,\ldots,k_{p(\lambda)} \leftarrow \{0,1\}^\lambda \\ |\vartheta\rangle \leftarrow \mathcal{H}_{n(\lambda)}}} \left[A_\lambda \left(\bigotimes_{i=1}^{p(\lambda)} G_{k_i}(|\vartheta\rangle)^{\otimes \ell(\lambda)} \otimes |\vartheta\rangle\langle\vartheta|^{\otimes t(\lambda)} \right) = 1 \right] \right.$$

$$\left. - \Pr_{\substack{|\varphi_1\rangle,\ldots,|\varphi_{p(\lambda)}\rangle \leftarrow \mathcal{H}_{n(\lambda)} \\ |\vartheta\rangle \leftarrow \mathcal{H}_{n(\lambda)}}} \left[A_\lambda \left(\bigotimes_{i=1}^{p(\lambda)} |\varphi_i\rangle\langle\varphi_i|^{\otimes \ell(\lambda)} \otimes |\vartheta\rangle\langle\vartheta|^{\otimes t(\lambda)} \right) = 1 \right] \right| \leq \mathsf{negl}(\lambda).$$

If G satisfies multi-key ℓ-copy pseudorandomness for every polynomial $\ell(\cdot)$ then we drop ℓ from the notation and simply denote it to be a multi-key (λ, n)-PRSG.

Remark 1. Note that in the plain model, PRS implies multi-key PRS because the pseudorandom state generator does not share randomness for different keys. It is not clear whether this holds in the CHS model as the different executions of the pseudorandom state generator share the same common Haar state.

Pseudorandom Function-like State (PRFS) Generators in the CHS Model

Definition 3 (Statistical Selectively Secure (λ, m, n, ℓ)-PRFS Generators). *We say that a QPT algorithm G is a* statistical selectively secure (λ, m, n, ℓ)-PRFS generator *in the CHS model if the following holds:*

- ***State Generation**: For any $\lambda \in \mathbb{N}$, $k \in \{0,1\}^\lambda$ and $x \in \{0,1\}^{m(\lambda)}$, where $m(\lambda)$ is the input length, the algorithm $G_{k,x}$ (where $G_{k,x}$ denotes $G(k, x, \cdot)$) is a quantum channel such that for every $n(\lambda)$-qubit state $|\vartheta\rangle$,*

$$G_{k,x}(|\vartheta\rangle\langle\vartheta|) = |\vartheta_{k,x}\rangle\langle\vartheta_{k,x}|,$$

  for some $n(\lambda)$-qubit state $|\vartheta_{k,x}\rangle$. We sometimes write $G_{k,x}(|\vartheta\rangle)$ or $G_k(x, |\vartheta\rangle)$ for brevity.

– ***ℓ-query Selective Security***: *For any polynomial $t(\cdot)$, any non-uniform, unbounded adversary $A = \{A_\lambda\}_{\lambda\in\mathbb{N}}$, and any tuple of (possibly repeated) $m(\lambda)$-bit indices $(x_1, \ldots, x_{\ell(\lambda)})$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$,*

$$\left| \Pr_{k\leftarrow\{0,1\}^\lambda, |\vartheta\rangle\leftarrow\mathcal{H}_{n(\lambda)}} \left[A_\lambda \left(x_1, \ldots, x_{\ell(\lambda)}, \bigotimes_{i=1}^{\ell(\lambda)} G(k, x_i, |\vartheta\rangle) \otimes |\vartheta\rangle\langle\vartheta|^{\otimes t(\lambda)} \right) = 1 \right] \right.$$
$$\left. - \Pr_{\substack{\forall x\in\{0,1\}^{m(\lambda)}, |\varphi_x\rangle\leftarrow\mathcal{H}_{n(\lambda)}, \\ |\vartheta\rangle\leftarrow\mathcal{H}_{n(\lambda)}}} \left[A_\lambda \left(x_1, \ldots, x_{\ell(\lambda)}, \bigotimes_{i=1}^{\ell(\lambda)} |\varphi_{x_i}\rangle\langle\varphi_{x_i}| \otimes |\vartheta\rangle\langle\vartheta|^{\otimes t(\lambda)} \right) = 1 \right] \right| \leq \mathsf{negl}(\lambda).$$

If G satisfies ℓ-query selective security for every polynomial $\ell(\cdot)$, we drop ℓ from the notation and say that G is a (λ, m, n)-PRFS generator.

Quantum Commitments in the CHS Model

Definition 4 (Quantum Commitments in the CHS Model). *A (non-interactive) quantum commitment scheme in the CHS model is given by a tuple of the committer C and receiver R parameterized by a polynomial $p(\cdot)$, both of which are uniform QPT algorithms. Let $|\vartheta\rangle$ be the $n(\lambda)$-qubit common Haar state. The scheme is divided into two phases: the commit phase, and the reveal phase as follows:*

– *Commit phase: C takes $|\vartheta\rangle^{\otimes p(\lambda)}$ and a bit $b \in \{0,1\}$ to commit as input, generates a quantum state on registers C and R, and sends the register C to R.*
– *Reveal phase: C sends b and the register R to R. R takes $|\vartheta\rangle^{\otimes p(\lambda)}$ and $(b, \mathsf{C}, \mathsf{R})$ given by C as input, and outputs b if it accepts and otherwise outputs $\bot$.*

Definition 5 (Poly-Copy Statistical Hiding). *A quantum commitment scheme (C, R) in the CHS model satisfies* poly-copy statistical hiding *if for any non-uniform, unbounded malicious receiver $R^* = \{R^*_\lambda\}_{\lambda\in\mathbb{N}}$, and any polynomial $t(\cdot)$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that*

$$\left| \Pr\left[R^*_\lambda(|\vartheta\rangle^{\otimes t(\lambda)}, \mathrm{Tr}_{\mathsf{R}}(\sigma_{\mathsf{CR}})) = 1 : \substack{|\vartheta\rangle\leftarrow\mathcal{H}_{n(\lambda)}, \\ \sigma_{\mathsf{CR}}\leftarrow C_{,}(|\vartheta\rangle^{\otimes p(\lambda)}, 0)} \right] \right.$$
$$\left. - \Pr\left[R^*_\lambda(|\vartheta\rangle^{\otimes t(\lambda)}, \mathrm{Tr}_{\mathsf{R}}(\sigma_{\mathsf{CR}})) = 1 : \substack{|\vartheta\rangle\leftarrow\mathcal{H}_{n(\lambda)}, \\ \sigma_{\mathsf{CR}}\leftarrow C_{,}(|\vartheta\rangle^{\otimes p(\lambda)}, 1)} \right] \right| \leq \mathsf{negl}(\lambda),$$

where $C_{,}$ is the commit phase of C.

Definition 6 (Statistical Sum-Binding). *A quantum commitment scheme (C, R) in the CHS model satisfies* statistical sum-binding *if the following holds.*

For any pair of non-uniform, unbounded malicious senders C_0^* and C_1^* that take $|\vartheta\rangle^{\otimes T(\lambda)}$ for arbitrary large $T(\cdot)$ as input and work in the same way in the commit phase, if we let p_b to be the probability that R accepts the revealed bit b in the interaction with C_b^* for $b \in \{0,1\}$, then we have

$$p_0 + p_1 \leq 1 + \mathsf{negl}(\lambda).$$

3.3 Symmetric Subspaces, Type States, and Haar States

The proofs of facts and lemmas stated in this subsection can be found in [Har13]. Let $\mathbf{v} = (v_1, \ldots, v_t) \in A^t$ for some finite set A. Let $|A| = N$. Define $\mathsf{type}(\mathbf{v}) \in [0:t]^N$ to be the *type vector* such that the i^{th} entry of $\mathsf{type}(\mathbf{v})$ equals the number of occurrences of $i \in [N]$ in $\mathbf{v}$.[11] In this work, by $T \in [0:t]^N$ we implicitly assume that $\sum_{i\in[N]} T_i = t$. For $T \in [0:t]^N$, we denote by $\mathsf{mset}(T)$ the *multiset* uniquely determined by T. That is, the multiplicity of $i \in \mathsf{mset}(T)$ equals T_i for all $i \in [N]$. We write $T \leftarrow [0:t]^N$ to mean sampling T uniformly from $[0:t]^N$ conditioned on $\sum_{i\in[N]} T_i = t$. We write $\mathbf{v} \in T$ to mean $\mathbf{v} \in A^t$ satisfies $\mathsf{type}(\mathbf{v}) = T$.

In this work, we will focus on *collision-free* types T which satisfy $T_i \in \{0,1\}$ for all $i \in [N]$. A collision-free type T can be naturally treated as a *set* and we write $\mathbf{v} \leftarrow T$ to mean sampling a uniform $\mathbf{v}$ conditioned on $\mathsf{type}(\mathbf{v}) = T$.

Definition 7 (Type States). *Let $T \in [0:t]^N$, we define the* type states*:*

$$|T\rangle := \sqrt{\frac{\prod_{i\in[N]} T_i!}{t!}} \sum_{\mathbf{v}\in T} |\mathbf{v}\rangle.$$

If T is collision-free, then it can be simplified to

$$|T\rangle = \frac{1}{\sqrt{t!}} \sum_{\mathbf{v}\in T} |\mathbf{v}\rangle.$$

Furthermore, it has the following useful expression

$$|T\rangle\langle T| = \frac{1}{t!} \sum_{\mathbf{v},\mathbf{u}\in T} |\mathbf{v}\rangle\langle\mathbf{u}| = \mathop{\mathbb{E}}_{\mathbf{v}\leftarrow T} \left[\sum_{\sigma\in S_t} |\mathbf{v}\rangle\langle\sigma(\mathbf{v})| \right]. \tag{1}$$

Lemma 1 (Average of Copies of Haar-Random States). *For all $N, t \in \mathbb{N}$, we have*

$$\mathop{\mathbb{E}}_{|\vartheta\rangle\leftarrow\mathcal{H}(\mathbb{C}^N)} |\vartheta\rangle\langle\vartheta|^{\otimes t} = \mathop{\mathbb{E}}_{T\leftarrow[0:t]^N} |T\rangle\langle T|.$$

[11] We identify $[0:t]^N$ as $[0:t]^A$.

3.4 Quantum Black-Box Reductions

We recall the definition of fully black-box reductions [RTV04, BBF13] and their quantum analogue. The definitions below are taken verbatim from [HY20].

Definition 8 (Quantum Primitives). *A quantum primitive $\mathcal{P}$ is a pair $(\mathcal{F}_\mathcal{P}, \mathcal{R}_\mathcal{P})$, where $\mathcal{F}_\mathcal{P}$ is a set of quantum algorithms $\mathcal{I}$, and $\mathcal{R}_\mathcal{P}$ is a relation over pairs $(\mathcal{I}, \mathcal{A})$ of quantum algorithms $\mathcal{I} \in \mathcal{F}_\mathcal{P}$ and $\mathcal{A}$. A quantum algorithm $\mathcal{I}$ implements $\mathcal{P}$ or is an implementation of $\mathcal{P}$ if $\mathcal{I} \in \mathcal{F}_\mathcal{P}$. If $\mathcal{I} \in \mathcal{F}_\mathcal{P}$ is efficient, then $\mathcal{I}$ is an efficient implementation of $\mathcal{P}$. A quantum algorithm $\mathcal{A}$ $\mathcal{P}$-breaks $\mathcal{I} \in \mathcal{F}_\mathcal{P}$ if $(\mathcal{I}, \mathcal{A}) \in \mathcal{R}_\mathcal{P}$. A secure implementation of $\mathcal{P}$ is an implementation $\mathcal{I}$ of $\mathcal{P}$ such that no efficient quantum algorithm $\mathcal{P}$-breaks $\mathcal{I}$. The primitive $\mathcal{P}$ quantumly exists if there exists an efficient and secure implementation of $\mathcal{P}$.*

Definition 9 (Quantum Primitives Relative to Oracle). *Let $\mathcal{P} = (\mathcal{F}_\mathcal{P}, \mathcal{R}_\mathcal{P})$ be a quantum primitive, and O be a quantum oracle. An oracle quantum algorithm $\mathcal{I}$ implements $\mathcal{P}$ relative to O or is an implementation of $\mathcal{P}$ relative to O if $\mathcal{I}^O \in \mathcal{F}_\mathcal{P}$. If $\mathcal{I}^O \in \mathcal{F}_\mathcal{P}$ is efficient, then $\mathcal{I}$ is an efficient implementation of $\mathcal{P}$ relative to O. A quantum algorithm $\mathcal{A}$ $\mathcal{P}$-breaks $\mathcal{I} \in \mathcal{F}_\mathcal{P}$ relative to O if $(I^O, \mathcal{A}^O) \in \mathcal{R}_\mathcal{P}$. A secure implementation of $\mathcal{P}$ is an implementation $\mathcal{I}$ of $\mathcal{P}$ relative to O such that no efficient quantum algorithm $\mathcal{P}$-breaks $\mathcal{I}$ relative to O. The primitive $\mathcal{P}$ quantumly exists relative to O if there exists an efficient and secure implementation of $\mathcal{P}$ relative to O.*

Definition 10 (Quantum Fully Black-Box Reductions). *A pair (C, S) of efficient oracle quantum algorithms is a* quantum fully-black-box reduction *from a quantum primitive $\mathcal{P} = (\mathcal{F}_\mathcal{P}, \mathcal{R}_\mathcal{P})$ to a quantum primitive $\mathcal{Q} = (\mathcal{F}_\mathcal{Q}, \mathcal{R}_\mathcal{Q})$ if the following two conditions are satisfied:*

1. ***(Correctness.)** For every implementation $\mathcal{I} \in \mathcal{F}_\mathcal{Q}$, we have $C^\mathcal{I} \in \mathcal{F}_\mathcal{P}$.*
2. ***(Security.)** For every implementation $\mathcal{I} \in \mathcal{F}_\mathcal{Q}$ and every quantum algorithm $\mathcal{A}$, if $\mathcal{A}$ $\mathcal{P}$-breaks $C^\mathcal{I}$, then $S^{\mathcal{A},\mathcal{I}}$ $\mathcal{Q}$-breaks $\mathcal{I}$.*

4 Warmup: Statistical Stretch PRS Generators inthe CHS Model

We present a construction of multi-key PRS generator with statistical security in the CHS model.

Theorem 7. *There exists a multi-key (λ, n, ℓ)-statistical PRS generator in the CHS model, where $n \geq \lambda$ and $\ell = O(\lambda/\log(\lambda)^{1+\varepsilon})$ for any constant $\varepsilon > 0$.*

The proof can be found in Sect. 4.2. Later, we prove the optimality of our construction in Sect. 4.3. Specifically, we show that any (λ, n, ℓ)-statistical PRS generator cannot simultaneously satisfy $n = \omega(\log(\lambda))$ and $\ell = \Omega(\lambda/\log(\lambda))$.

4.1 Useful Lemmas

At a high level, the proof follows the template of [AGQY22,AGKL23]: we do the analysis in the symmetric subspace. First, we identify a nice property of type vectors such that (1) a randomly sampled type satisfies this property with overwhelming probability and (2) the PRS generation algorithm behaves well on every type state having this property. We identify these type vectors as *ℓ-fold collision-free* types (which are a generalization of distinct types [AGQY22, AGKL23]).

Definition 11 (ℓ-Fold n-Prefix Collision-Free Types). *Let $n, m, t, \ell \in \mathbb{N}$ such that $t \geq \ell$ and $T \in [0:t]^{2^{n+m}}$ is a type vector. We say that T is* ℓ-fold n-prefix collision-free *if for all pairs of ℓ-subsets[12] $\mathcal{S}, \mathcal{T} \subseteq \mathsf{mset}(T)$, the first n bits of $\bigoplus_{x \in \mathcal{S}} x \in \{0,1\}^{n+m}$ is identical to that of $\bigoplus_{y \in \mathcal{T}} y \in \{0,1\}^{n+m}$ if and only if $\mathcal{S} = \mathcal{T}$. We define $\mathcal{I}^{(\ell)}_{n,m}(t) := \{T \in [0:t]^{2^{n+m}} : T$ is ℓ-fold n-prefix collision-free$\}$ as the set of all ℓ-fold n-prefix collision-free type vectors.*

When $t > \ell$, one can easily verify that ℓ-fold n-prefix collision-freeness implies the standard collision-freeness. Also note that when $t > 2\ell$, ℓ-fold n-prefix collision-freeness implies i-fold n-prefix collision-freeness for all $i \leq \ell$.

Next, we show that a random type is ℓ-fold n-prefix collision-free with high probability.

Lemma 2. $\Pr_{T \leftarrow [0:t]^{2^{n+m}}}[T \in \mathcal{I}^{(\ell)}_{n,m}(t)] = 1 - O(t^{2\ell}/(2^n - 2\ell))$.

Proof. First, sampling $T \leftarrow [0:t]^{2^{n+m}}$ uniformly is $O(t^2/2^{n+m})$-close to sampling a uniform collision-free T from $[0:t]^{2^{n+m}}$ by the collision bound.
Furthermore, sampling a uniform collision-free T from $[0:t]^{2^{n+m}}$ is equivalent to sampling t elements $x_1, x_2, \ldots, x_t$ one by one from $\{0,1\}^{n+m}$ conditioned on them being distinct and setting T such that $\mathsf{mset}(T) = \{x_1, \ldots, x_t\}$. Hence, it suffices to show that sampling t elements $x_1, x_2, \ldots, x_t$ one by one from $\{0,1\}^{n+m}$ conditioned on them being distinct results in an ℓ-fold n-prefix collision-free set with probability $1 - O(t^{2\ell}/2^n)$.
For any two distinct ℓ-subsets of indices $\mathcal{S} \neq \mathcal{T} \subseteq [t]$, let $\mathsf{Bad}_{\mathcal{S},\mathcal{T}}$ denote the event that the first n bits of $\bigoplus_{i \in \mathcal{S}} x_i$ is the same as that of $\bigoplus_{j \in \mathcal{T}} x_j$. Then the following holds:

$$\Pr\left[\mathsf{Bad}_{\mathcal{S},\mathcal{T}} : \begin{smallmatrix} x_1,x_2,\ldots,x_t \leftarrow \{0,1\}^{n+m} \\ x_1,x_2,\ldots,x_t \text{ are distinct} \end{smallmatrix}\right] = O(1/(2^n - 2\ell)).$$

This is because we can first sample $|\mathcal{S} \cup \mathcal{T}| - 1$ elements (in $\mathcal{S} \cup \mathcal{T}$) except one with indices in $\mathcal{S} \setminus \mathcal{T}$. Then $\mathsf{Bad}_{\mathcal{S},\mathcal{T}}$ occurs only if the first n bits of the last sample is equal to the first n bits of the bitwise XOR of all other elements in $\mathcal{S}$ with all elements in $\mathcal{T}$, which happens with probability at most $O(1/(2^n - 2\ell))$.

By a union bound, we have $T \in \mathcal{I}^{(\ell)}_{n,m}(t)$ with probability at least $1 - (O(t^2/2^{n+m}) + \binom{t}{\ell}^2 \cdot O(1/(2^n - 2\ell))) = 1 - O(t^{2\ell}/(2^n - 2\ell))$. □

[12] Here we allow the subsets to contain duplicate elements.

Finally, the following two lemmas show that applying random Pauli-Z on any ℓ-fold n-prefix collision-free type state is equivalent to a "classical" probabilistic process[13].

Lemma 3. *For any* $\mathbf{v} \in \{0,1\}^{(n+m)(t+\ell)}$ *such that* $\mathsf{type}(\mathbf{v}) \in \mathcal{I}^{(\ell)}_{n,m}(t+\ell)$ *and* $\sigma \in S_{t+\ell}$*, define*

$$A_{\mathbf{v},\sigma} := \mathop{\mathbb{E}}_{k \leftarrow \{0,1\}^n} \left[\left(\left(Z^k \otimes I_m\right)^{\otimes \ell} \otimes I_{n+m}^{\otimes t} \right) |\mathbf{v}\rangle\langle\sigma(\mathbf{v})| \left(\left(Z^k \otimes I_m\right)^{\otimes \ell} \otimes I_{n+m}^{\otimes t} \right) \right].$$

Then $A_{\mathbf{v},\sigma} = |\mathbf{v}\rangle\langle\sigma(\mathbf{v})|$ *if* σ *maps* $[\ell]$ *to* $[\ell]$*; otherwise,* $A_{\mathbf{v},\sigma} = 0$.

Proof. Suppose $\mathbf{v} = (v_1||w_1, \ldots, v_{t+\ell}||w_{t+\ell}) \in \{0,1\}^{(n+m)(t+\ell)}$ with $v_i \in \{0,1\}^n$ and $w_i \in \{0,1\}^m$ for all $i \in [t]$. First, a direct calculation yields:

$$\begin{aligned}
&\left(\left(Z^k \otimes I_m\right)^{\otimes \ell} \otimes I_{n+m}^{\otimes t} \right) |\mathbf{v}\rangle\langle\sigma(\mathbf{v})| \left(\left(Z^k \otimes I_m\right)^{\otimes \ell} \otimes I_{n+m}^{\otimes t} \right) \\
&= (-1)^{\langle k, \bigoplus_{i=1}^{\ell} (v_i \oplus v_{\sigma(i)}) \rangle} |\mathbf{v}\rangle\langle\sigma(\mathbf{v})|.
\end{aligned}$$

Therefore, after averaging over k,

$$\begin{aligned}
A_{\mathbf{v},\sigma} &= \mathop{\mathbb{E}}_{k \leftarrow \{0,1\}^n} \left[(-1)^{\langle k, \bigoplus_{i=1}^{\ell} (v_i \oplus v_{\sigma(i)}) \rangle} \right] |\mathbf{v}\rangle\langle\sigma(\mathbf{v})| \\
&= \begin{cases} |\mathbf{v}\rangle\langle\sigma(\mathbf{v})| & \text{if } \bigoplus_{i=1}^{\ell} (v_i \oplus v_{\sigma(i)}) = 0 \\ 0 & \text{otherwise.} \end{cases}
\end{aligned}$$

Since $\mathsf{type}(\mathbf{v}) \in \mathcal{I}^{(\ell)}_{n,m}(t+\ell)$, the condition $\bigoplus_{i=1}^{\ell} v_i = \bigoplus_{i=1}^{\ell} v_{\sigma(i)}$ holds if and only if the two sets $\{1, 2, \ldots, \ell\}$ and $\{\sigma(1), \sigma(2), \ldots, \sigma(\ell)\}$ are identical. □

The following lemma lies at the technical heart of this section. It states that the action of applying random Z^k on ℓ-fold n-prefix collision-free types T[14] has the following "classical" probabilistic interpretation: the output is identically distributed to first uniformly sampling an ℓ-subset X from T and then generating $|X\rangle\langle X| \otimes |T \setminus X\rangle\langle T \setminus X|$.

Lemma 4. *For any* $T \in \mathcal{I}^{(\ell)}_{n,m}(t+\ell)$*,*

$$\begin{aligned}
&\mathop{\mathbb{E}}_{k \leftarrow \{0,1\}^n} \left[\left(\left(Z^k \otimes I_m\right)^{\otimes \ell} \otimes I_{n+m}^{\otimes t} \right) |T\rangle\langle T| \left(\left(Z^k \otimes I_m\right)^{\otimes \ell} \otimes I_{n+m}^{\otimes t} \right) \right] \\
&= \mathop{\mathbb{E}}_{X \leftarrow \binom{T}{\ell}} \left[|X\rangle\langle X| \otimes |T \setminus X\rangle\langle T \setminus X| \right].
\end{aligned}$$

[13] We say that this is a "classical" probabilistic process because we can write the resulting density matrix as direct sum of matrices with classical descriptions with weights chosen by a completely classical process. This means that we can simualte this process by first doing a completely classical sampling process followed by a state preparation.

[14] Since T is collision-free, we will treat it as a set.

Proof. We first use the expression in Eq. (1) on the left-hand side:

$$\mathop{\mathbb{E}}_{\mathbf{v}\leftarrow T}\left[\sum_{\sigma\in S_t}\mathop{\mathbb{E}}_{k\leftarrow\{0,1\}^n}\left[\left(\left(Z^k\otimes I_m\right)^{\otimes\ell}\otimes I_{n+m}^{\otimes t}\right)|\mathbf{v}\rangle\langle\sigma(\mathbf{v})|\left(\left(Z^k\otimes I_m\right)^{\otimes\ell}\otimes I_{n+m}^{\otimes t}\right)\right]\right]. \tag{2}$$

Then from the previous lemma (Lemma 3)

$$\begin{aligned}
(2) &= \mathop{\mathbb{E}}_{\mathbf{v}\leftarrow T}\left[\sum_{\sigma_1\in S_\ell,\sigma_2\in S_t}|\mathbf{v}\rangle\langle\sigma_1\circ\sigma_2(\mathbf{v})|\right]\\
&= \mathop{\mathbb{E}}_{\mathbf{v}\leftarrow T}\left[\sum_{\sigma_1\in S_\ell}|\mathbf{v}_{[1:\ell]}\rangle\langle\sigma_1(\mathbf{v}_{[1:\ell]})|\otimes\sum_{\sigma_2\in S_t}|\mathbf{v}_{[\ell+1:\ell+t]}\rangle\langle\sigma_2(\mathbf{v}_{[\ell+1:\ell+t]})|\right]\\
&= \mathbb{E}\left[\sum_{\sigma_1\in S_\ell}|\mathbf{v}_1\rangle\langle\sigma_1(\mathbf{v}_1)|\otimes\sum_{\sigma_2\in S_t}|\mathbf{v}_2\rangle\langle\sigma_2(\mathbf{v}_2)|:\begin{smallmatrix}X\leftarrow\binom{T}{\ell},\\ \mathbf{v}_1\leftarrow X,\\ \mathbf{v}_2\leftarrow T\setminus X\end{smallmatrix}\right]\\
&= \mathop{\mathbb{E}}_{X\leftarrow\binom{T}{\ell}}\left[|X\rangle\langle X|\otimes|T\setminus X\rangle\langle T\setminus X|\right].
\end{aligned}$$

For the first equality, we use Lemma 3 and decompose $\sigma=\sigma_1\circ\sigma_2$ for some σ_1,σ_2 such that $\sigma_1(x)=x$ for all $x\in\{\ell+1,\ell+2,\cdots,\ell+t\}$ and $\sigma_2(y)=y$ for all $y\in\{1,2,\cdots,\ell\}$. Since all $\ell+1,\ell+2,\cdots,\ell+t$ are fixed points of σ_1, we can view it as an element in S_ℓ. Similarly, we view $\sigma_2(y)$ as an element in S_t. The second equality follows by denoting the first ℓ part of $\mathbf{v}$ by $\mathbf{v}_{[1:\ell]}$ and the last t part of $\mathbf{v}$ by $\mathbf{v}_{[\ell+1:\ell+t]}$. The third equality holds because sampling a tuple $\mathbf{v}$ from T is equivalent to sampling an ℓ-subset X from T followed by ordering to elements in X and $T\setminus X$. □

4.2 Construction

In this section, we assume that the length of the common Haar state satisfies $n=n(\lambda)\geq\lambda$ for all $\lambda\in\mathbb{N}$. We define the construction as follows: on input $k\in\{0,1\}^\lambda$ and a single copy of the common Haar state $|\vartheta\rangle$,

$$G_k(|\vartheta\rangle):=(Z^k\otimes I_{n-\lambda})|\vartheta\rangle.$$

Lemma 5 (ℓ-Copy Pseudorandomness). *Let G be as defined above. Let*

$$\rho:=\mathop{\mathbb{E}}_{\substack{k\leftarrow\{0,1\}^\lambda\\ |\vartheta\rangle\leftarrow\mathcal{H}_n}}\left[G_k(|\vartheta\rangle)^{\otimes\ell}\otimes|\vartheta\rangle\langle\vartheta|^{\otimes t}\right]\ \text{and}\ \sigma:=\mathop{\mathbb{E}}_{\substack{|\varphi\rangle\leftarrow\mathcal{H}_n\\ |\vartheta\rangle\leftarrow\mathcal{H}_n}}\left[|\varphi\rangle\langle\varphi|^{\otimes\ell}\otimes|\vartheta\rangle\langle\vartheta|^{\otimes t}\right].$$

Then $\mathsf{TD}(\rho,\sigma)=O\left(\frac{(\ell+t)^{2\ell}}{2^\lambda}\right)$.

Proof. We prove this via a hybrid argument:

Hybrid 1. Sample $T \leftarrow [0:\ell+t]^{2^n}$. Sample $k \leftarrow \{0,1\}^\lambda$. Output $((Z^k \otimes I_{n-\lambda})^{\otimes \ell} \otimes I_n^{\otimes t})|T\rangle$.

Hybrid 2. Sample $T \leftarrow [0:\ell+t]^{2^n}$ uniformly conditioned on $T \in \mathcal{I}^{(\ell)}_{\lambda,n-\lambda}(\ell+t)$. Sample $k \leftarrow \{0,1\}^\lambda$. Output $((Z^k \otimes I_{n-\lambda})^{\otimes \ell} \otimes I_n^{\otimes t})|T\rangle$.

Hybrid 3: Sample $T \leftarrow [0:\ell+t]^{2^n}$ uniformly conditioned on $T \in \mathcal{I}^{(\ell)}_{\lambda,n-\lambda}(\ell+t)$. Sample a uniform ℓ-subset T_1 from T. Output $|T_1\rangle \otimes |T \setminus T_1\rangle$.

Hybrid 4. Sample $T \leftarrow [0:\ell+t]^{2^n}$. Sample a uniform ℓ-subset T_1 from T.[15] Output $|T_1\rangle \otimes |T \setminus T_1\rangle$.

Hybrid 5. Sample a collision-free T from $[0:\ell+t]^{2^n}$. Sample a uniform ℓ-subset T_1 from T. Output $|T_1\rangle \otimes |T \setminus T_1\rangle$.

Hybrid 6. Sample a uniform collision-free T_1 from $[0:\ell]^{2^n}$. Sample a uniform collision-free T_2 from $[0:t]^{2^n}$ conditioned on T_1 and T_2 have no common elements. Output $|T_1\rangle \otimes |T_2\rangle$.

Hybrid 7. Sample a uniform collision-free T_1 from $[0:\ell]^{2^n}$. Sample a uniform collision-free T_2 from $[0:t]^{2^n}$. Output $|T_1\rangle \otimes |T_2\rangle$.

Hybrid 8. Sample $T_1 \leftarrow [0:\ell]^{2^n}$. Sample $T_2 \leftarrow [0:t]^{2^n}$. Output $|T_1\rangle \otimes |T_2\rangle$.

Indistinuishability of Hybrids.

- By Lemma 2, the trace distance between Hybrid 1 and Hybrid 2 is $O((t+\ell)^{2\ell}/2^\lambda)$.
- From Lemma 4, the output of Hybrid 2 is

$$\mathop{\mathbb{E}}_{\substack{T \leftarrow [0:\ell+t]^{2^n}: \\ T \in \mathcal{I}^{(\ell)}_{\lambda,n-\lambda}(\ell+t)}} \; \mathop{\mathbb{E}}_{T_1 \leftarrow \binom{T}{\ell}} \left[|T_1\rangle\langle T_1| \otimes |T \setminus T_1\rangle\langle T \setminus T_1|\right].$$

  Hence, Hybrid 2 is equivalent to Hybrid 3.
- Again by Lemma 2, the trace distance between Hybrid 3 and Hybrid 4 is $O((t+\ell)^{2\ell}/2^\lambda)$.
- The trace distance between Hybrid 4 and Hybrid 5 is $O((t+\ell)^2/2^n)$ by the collision bound.
- Hybrid 5 and Hybrid 6 are equivalent.
- The trace distance between Hybrid 6 and Hybrid 7 is $O(t\ell/2^n)$.

[15] Since T might have collisions, T_1 is allowed to contain duplicate elements.

– Finally, the trace distance between Hybrid 7 and Hybrid 8 is $O((t^2+\ell^2)/2^n)$ by the collision bound.

This completes the proof. □

In the following, we show that our construction also satisfies multi-key ℓ-copy pseudorandomness using Lemma 5.

Lemma 6 (Multi-key ℓ-Copy Pseudorandomness). *Let G be defined as above. Let*

$$\rho := \bigotimes_{i=1}^{p} \mathop{\mathbb{E}}_{|\varphi_i\rangle \leftarrow \mathcal{H}_n} \left[|\varphi_i\rangle\langle\varphi_i|^{\otimes \ell} \right] \otimes \mathop{\mathbb{E}}_{|\vartheta\rangle \leftarrow \mathcal{H}_n} \left[|\vartheta\rangle\langle\vartheta|^{\otimes t} \right]$$

and

$$\sigma := \mathop{\mathbb{E}}_{|\vartheta\rangle \leftarrow \mathcal{H}_n} \left[\bigotimes_{i=1}^{p} \mathop{\mathbb{E}}_{k_i \leftarrow \{0,1\}^\lambda} \left[G_{k_i}(|\vartheta\rangle)^{\otimes \ell} \right] \otimes |\vartheta\rangle\langle\vartheta|^{\otimes t} \right].$$

Then $\mathsf{TD}(\rho,\sigma) = O\left(\frac{p\cdot(p\ell+t)^{2\ell}}{2^\lambda}\right)$.

The proof of Lemma 6 can be found in the full version [AGL24].

Proof of Theorem 7. Our construction is an efficiently-implementable unitary channel and thus satisfies the state generation property. Pseudorandomness follows from Lemma 6. □

4.3 Optimality of Our PRSG Construction

In this section, if the PRS generation algorithm uses only *one copy* of the common Haar state, we show that ℓ-copy statistical PRS and multi-key ℓ-copy statistical PRS are impossible for $\ell = \Omega(\lambda/\log(\lambda))$ and $n = \omega(\log(\lambda))$.

Theorem 8. *Statistically secure (λ, n, ℓ)-PRS is impossible in the CHS model if (a) the generation algorithm uses only one copy of the common Haar state, (b) $n = \omega(\log(\lambda))$, (c) $\ell = \Omega(\lambda/\log(\lambda))$ and, (d) the length of the common Haar state is $n = \omega(\log(\lambda))$.*

The proof of Theorem 8 can be found in the full version [AGL24].

5 Statistical Stretch PRFS Generators in the CHS Model

In this section, we extend our techniques from Sect. 4.2 to construct an (λ, m, n, ℓ)-statistical PRFS in the CHS model, where $m = \lambda^c$, $\ell = \lambda^{1-c}/\log(\lambda)^{1+\varepsilon}$, the length of the common Haar state is $n \geq \lambda^{1-c}$, for any constant $\varepsilon > 0$ and $c \in [0,1)$. In the case when $n > \lambda$, the construction satisfies stretch property. We prove the following theorem in the full version [AGL24].

Theorem 9. *There exists an (λ, m, n, ℓ)-statistical selectively secure PRFS generator in the CHS model where the length of the common Haar state is $n(\lambda)$, $m(\lambda) = \lambda^c$, $\ell = O(\lambda^{1-c}/\log(\lambda)^{1+\varepsilon})$ and $n(\lambda) \geq \lambda^{1-c}$, for any constant $\varepsilon > 0$ and for any $c \in [0,1)$.*

Note that since a PRS can be used to computationally instantiate CHS in the plain model, the above result also gives us a way to get bounded-query long-input PRFS from PRS in the plain model. In more detail, we can start with a PRS that has stretch (i.e. $n > \lambda$) and then we can bootstrap into a PRFS for large input length at the cost of a reduction in stretch.[16]

Corollary 2. Assuming the existence of (λ, n, ℓ)-PRS, for $n > \lambda$ and $\ell = O(\lambda^{1-c}/\log(\lambda)^{1+\varepsilon})$, there exists a selectively secure $(2\lambda, m, n, \ell)$-PRFS generator with $m(\lambda) = \lambda^c$, for any constant $\varepsilon > 0$ and for any $c \in [0, 1)$.

Furthermore, since PRFS imply PRS, achieving an ℓ-query statistical PRFS in the CHS model for $\ell = \Omega(\lambda/\log(\lambda))$ is impossible from Theorem 8.

Corollary 3. (λ, m, n, ℓ)-statistical PRFS is impossible in the CHS model if (a) the generation algorithm uses only one copy of the common Haar state, (b) $\ell = \Omega(\lambda/\log(\lambda))$, (c) the length of the common Haar state is n and, (d) $n = \omega(\log(\lambda))$.

5.1 Construction

We extend the techniques used in Sect. 4.2 to construct a statistical PRFS in Fig. 1. The construction samples a uniform key for each position of the input being zero or one. Applying this to the common Haar state gives us the output of the PRFS. The details can be seen in Fig. 1. Thoughout this section, one should think of $m = \lambda^c$ and $\lambda' = \lambda^{1-c}$ for some constant $c \in [0, 1)$.

Given the common Haar state $|\vartheta\rangle$, on the key $K = (k_1^0, \ldots, k_m^0, k_1^1, \ldots, k_m^1) \in \{0,1\}^{2\lambda' m}$ and the input $\mathbf{x} = (x_1, \ldots, x_m) \in \{0,1\}^m$, define $G(K, \mathbf{x}, |\vartheta\rangle)$ as follows:

- $|\psi_{K,\mathbf{x}}\rangle = G(K, \mathbf{x}, |\vartheta\rangle) = (Z^{\bigoplus_{i=1}^m k_i^{x_i}} \otimes I_{n-\lambda'})|\vartheta\rangle$.
- Output $|\psi_{K,\mathbf{x}}\rangle$.

Fig. 1. PRFS in the CHS model

The main property of the construction that makes it a PRFS is its ability to *disentangles* any type state in $\mathcal{I}^{(\ell)}_{\lambda', n-\lambda'}(\ell + t)$ into a probabilistic mixture of disjoint subsets of the type.

[16] Formally, let G_{PRS} is a (λ, n, ℓ)-PRS and $G(k, x, |\phi\rangle)$ is (λ, m, n, ℓ)-statistical selectively secure PRFS generator in the CHS model with $n > \lambda$, $\ell = O(\lambda^{1-c}/\log(\lambda)^{1+\varepsilon})$ and $m(\lambda) = \lambda^c$, then for $K = (k_1, k_2) \in \{0,1\}^\lambda \times \{0,1\}^\lambda$ we can define $G_{PRFS}(k, x) := G(k_1, x, G_{PRS}(k_2))$ as the $(2\lambda, m, n, \ell)$-PRFS generator.

6 Quantum Commitments in the CHS Model

In this section, we construct a commitment scheme that satisfies poly-copy statistical hiding and statistical sum-biding in the CHS model. The scheme is inspired by the quantum commitment scheme proposed in [MY21, MNY23]. In contrast to the scheme in [MY21], our construction is not of the canonical form [Yan2]. To achieve binding, similar to [MNY23], the receiver needs to perform several SWAP tests. To achieve hiding, our scheme relies on the multi-key pseudorandomness property in Lemma 6.

6.1 Construction

We assume that $n(\lambda) \geq \lambda + 1$ for all $\lambda \in \mathbb{N}$. Our construction, parameterized by the polynomial $p = p(\lambda) := \lambda$, is shown in Fig. 2. In the full version [AGL24], we prove the following theorem:

Theorem 10. *The construction in Fig. 2 is a quantum commitment in the CHS model.*

Commit phase: The sender C_λ on input $b \in \{0,1\}$ does the following:

- Use p copies of the common Haar state $|\vartheta\rangle$ to prepare the state $|\Psi_b\rangle_{\mathsf{CR}} := \bigotimes_{i=1}^{p} |\psi_b\rangle_{\mathsf{C}_i\mathsf{R}_i}$, where

$$|\psi_0\rangle_{\mathsf{C}_i\mathsf{R}_i} := \frac{1}{\sqrt{2^\lambda}} \sum_{k\in\{0,1\}^\lambda} (Z^k \otimes I_{n-\lambda})|\vartheta\rangle_{\mathsf{C}_i}|k||0^{n-\lambda}\rangle_{\mathsf{R}_i}$$

  and

$$|\psi_1\rangle_{\mathsf{C}_i\mathsf{R}_i} := \frac{1}{\sqrt{2^n}} \sum_{j\in\{0,1\}^n} |j\rangle_{\mathsf{C}_i}|j\rangle_{\mathsf{R}_i},$$

  and $\mathsf{C} := (\mathsf{C}_1, \mathsf{C}_2, \ldots, \mathsf{C}_p)$ and $\mathsf{R} := (\mathsf{R}_1, \mathsf{R}_2, \ldots, \mathsf{R}_p)$.
- Send register C to the receiver.

Reveal phase:

- The sender sends b and register R to the receiver.
- The receiver prepares the state $|\Psi_b\rangle_{\mathsf{C}'\mathsf{R}'} = \bigotimes_{i=1}^{p} |\psi_b\rangle_{\mathsf{C}'_i\mathsf{R}'_i}$ by using p copies of the common Haar state $|\vartheta\rangle$, where $\mathsf{C}' := (\mathsf{C}'_1, \mathsf{C}'_2, \ldots, \mathsf{C}'_p)$ and $\mathsf{R}' := (\mathsf{R}'_1, \mathsf{R}'_2, \ldots, \mathsf{R}'_p)$ are receiver's registers.
- For $i \in [p]$, the receiver performs the SWAP test between registers $(\mathsf{C}_i, \mathsf{R}_i)$ and $(\mathsf{C}'_i, \mathsf{R}'_i)$.
- The receiver outputs b if all SWAP tests accept; otherwise, outputs $\perp$.

Fig. 2. Quantum commitment scheme in the CHS model

7 LOCC Indistinguishability

In this section, we prove our main technical theorem for proving impossibilities and separations in Sect. 8 and Sect. 9.

7.1 Definitions

Definition 12 (LOCC Adversaries). *An* LOCC adversary *is a tuple* (A, B)*, where* A *and* B *are spatially separated, non-uniform, and computationally unbounded quantum algorithms without pre-shared entanglement. In addition,* A *and* B *can only perform local operations on their registers and communicate classically.*

Definition 13 (LOCC Indistinguishability). *We say that two density matrices* $(\rho_{\mathsf{AB}}, \sigma_{\mathsf{AB}})$ *are* ε-LOCC indistinguishable *if for any LOCC adversary* (A, B) *with* A *taking as input register* A *and* B *taking as input register* B*, the probability that* B *outputs* 1 *satisfies*[17]

$$|\Pr[(A,B)(\rho_{\mathsf{AB}}) = 1] - \Pr[(A,B)(\sigma_{\mathsf{AB}}) = 1]| \leq \varepsilon.$$

If $\varepsilon(\cdot)$ *is negligible, then we simply say that* $(\rho_{\mathsf{AB}}, \sigma_{\mathsf{AB}})$ *are LOCC indistinguishable.*

7.2 LOCC Haar Indistinguishability

We prove the following theorem in the full version [AGL24]:

Theorem 11 (LOCC Haar Indistinguishability). *Let* $\rho_{\mathsf{AB}} := \mathbb{E}_{|\psi\rangle\leftarrow\mathcal{H}_n} |\psi\rangle\langle\psi|_{\mathsf{A}}^{\otimes t} \otimes |\psi\rangle\langle\psi|_{\mathsf{B}}^{\otimes t}$ *and* $\sigma_{\mathsf{AB}} := \mathbb{E}_{|\psi\rangle\leftarrow\mathcal{H}_n} \left[|\psi\rangle\langle\psi|_{\mathsf{A}}^{\otimes t}\right] \otimes \mathbb{E}_{|\phi\rangle\leftarrow\mathcal{H}_n} \left[|\phi\rangle\langle\phi|_{\mathsf{B}}^{\otimes t}\right]$*. Then* ρ_{AB} *and* σ_{AB} *are* $O(t^2/2^n)$*-LOCC indistinguishable.*

7.3 An Optimal LOCC Haar Distinguisher

We present an (optimal) LOCC Haar distinguisher with advantage $\Omega(t^2/2^n)$ in the full version [AGL24]. Hence, the upper bound in Theorem 11 is tight.

8 Impossibilities of QCCC Primitives in the CHS Model

In this section, we investigate the impossibility of *statistically* secure quantum-computation classical-communication (QCCC) primitives in the CHS model. We prove the following theorem in the full version [AGL24]:

Theorem 12. *There does not exist primitive* $\mathcal{P}$ *in the CHS model where* $\mathcal{P} \in$ *{statistically secure QCCC key agreements, statistically hiding and statistically binding QCCC interactive commitments}.*

[17] Since (A, B) are allowed to communicate and we do not care about communication complexity, it is without loss of generality to assume that B outputs the bit.

9 Quantum Black-Box Separation in the QCCC Model

9.1 The Separating Oracle

As is common in black-box impossibility results, we will define oracles relative to which $\omega(\log(\lambda))$-PRSGs exist while QCCC key agreements and interactive commitments do not. We define the oracle $G := \{\{G_k\}_{k\in\{0,1\}^\lambda}\}_{\lambda\in\mathbb{N}}$ as follows. For every $\lambda \in \mathbb{N}$ and $k \in \{0,1\}^\lambda$, the oracle G_k is a Haar isometry that maps any state $|\psi\rangle$ to $|\psi\rangle|\vartheta_k\rangle$, where $|\vartheta_k\rangle$ is a Haar state of length $n(\lambda) = \omega(\log(\lambda))$. The existence of $\omega(\log(\lambda))$-PRSGs relative to G can be proven easily.

9.2 Separating QCCC Key Agreements from $(\lambda, \omega(\log(\lambda)))$-PRSGs

In the full version [AGL24], we prove the following theorem:

Theorem 13. *There does not exist a quantum fully black-box reduction (C, S) from QCCC key agreements to $(\lambda, \omega(\log(\lambda)))$-PRSGs such that C only asks classical queries to the PRSG.*

9.3 Separating QCCC Interactive Commitments from $(\lambda, \omega(\log(\lambda)))$-PRSGs

In the full version [AGL24], we prove the following theorem:

Theorem 14. *There does not exist a quantum fully black-box reduction (C, S) from QCCC Interactive Commitments to $(\lambda, \omega(\log(\lambda)))$-PRSGs such that C only asks classical queries to the PRSG.*

Acknowledgements. This work is supported by the National Science Foundation under Grant No. 2329938 and Grant No. 2341004.

References

[ACC+22] Austrin, P., Chung, H., Chung, K.-M., Fu, S., Lin, Y.-T., Mahmoody, M.: On the impossibility of key agreements from quantum random oracles. In: Annual International Cryptology Conference, pp. 165–194. Springer, Cham (2022)

[ACH+23] Afshar, A., Chung, K.-M., Hsieh, Y.-C., Lin, Y.-T., Mahmoody, M.: On the (im) possibility of time- lock puzzles in the quantum random oracle model. In: International Conference on the Theory and Application of Cryptology and Information Security, pp. 339–368. Springer, Singapore (2023). (cit. on p. 6)

[AGKL23] Ananth, P., Gulati, A., Kaleoglu, F., Lin, Y.- T.: Pseudorandom Isometries. arXiv preprint arXiv:2311.02901 [quant-ph] (2023)

[AGL24] Ananth, P., Gulati, A., Lin, Y.-T.: Cryptography in the common Haar state model: feasibility results and separations. Cryptology ePrint Archive, Paper 2024/1043 (2024). https://eprint.iacr.org/2024/1043

[AGQY22] Ananth, P., Gulati, A., Qian, L., Yuen, H.: Pseudorandom (function-like) quantum state generators: new definitions and applications. In: Theory of Cryptography Conference, pp. 237–265. Springer, Cham (2022)

[AHY23] Ananth, P., Hu, Z., Yuen, H.: On the (im) plausibility of public-key quantum money from collision-resistant hash functions. In: International Conference on the Theory and Application of Cryptology and Information Security, pp. 39–72. Springer, Singapore (2023)

[AKY24] Ananth, P., Kaleoglu, F., Yuen, H.: Simultaneous Haar Indistinguishability with applications to unclonable cryptography. In: arXiv preprint arXiv:2405.10274 (2024)

[AQY22] Ananth, P., Qian, L., Yuen, H.: Cryptography from pseudorandom quantum states. In: CRYPTO (2022)

[BBF13] Baecher, P., Brzuska, C., Fischlin, M.: Notions of black-box reductions, revisited. In: Advances in Cryptology- ASIACRYPT 2013: 19th International Conference on the Theory and Application of Cryptology and Information Security, Bengaluru, 1–5 December 2013, Proceedings, Part I, 19, pp. 296–315. Springer, Heidelberg (2013)

[BCKM21] Bartusek, J., Coladangelo, A., Khurana, D., Ma, F.: One-way functions imply secure computation in a quantum world. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021. LNCS, vol. 12825, pp. 467–496. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-84242-0_17

[BCQ23] Brakerski, Z., Canetti, R., Qian, L.: On the computational hardness needed for quantum cryptography. In: 14th Innovations in Theoretical Computer Science Conference, ITCS 2023, p. 24. Schloss Dagstuhl-Leibniz-Zentrum fur Informatik GmbH, Dagstuhl Publishing (2023)

[BDF+99] Bennett, C.H., et al.: Quantum nonlocality without entanglement. Phys. Rev. A **59**(2), 1070 (1999)

[BFM19] Blum, M., Feldman, P., Micali, S.: Non-interactive zero-knowledge and its applications. In: Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali, pp. 329–349 (2019)

[BGVV+23] Bouaziz, S., Grilo, A.B., Vergnaud, D., Vu, Q.-H., et al.: Towards the impossibility of quantum public key encryption with classical keys from one-way functions. In: Cryptology ePrint Archive (2023)

[BL18] Benhamouda, F., Lin, H.: k-round multiparty computation from k-round oblivious transfer via garbled interactive circuits. In: Advances in Cryptology–EUROCRYPT 2018: 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, 29 April–3 May 2018, Part II 37, pp. 500–532. Springer, Cham (2018)

[BM+24] Bouaziz, S., Muguruza, G., et al.: Quantum Pseudorandomness Cannot be Shrunk in a Black-Box Way. In: Cryptology ePrint Archive (2024)

[Bra23] Brakerski, Z.: Black-hole radiation decoding is quantum cryptography. In: Annual International Cryptology Conference, pp. 37–65. Springer (2023)

[CCHL22] Chen, S., Cotler, J., Huang, H.-Y., Li, J.: Exponential separations between learning with and without quantum memory. In: 2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS), pp. 574–585. IEEE (2022)

[CCS24] Chen, B., Coladangelo, A., Sattath, O.: The power of a single Haar random state: constructing and separating quantum pseudorandomness. In: arXiv preprint arXiv:2404.03295 (2024)

[CF01] Canetti, R., Fischlin, M.: Universally composable commitments. In: Advances in Cryptology—CRYPTO 2001: 21st Annual International Cryptology Conference, Santa Barbara, 19–23 August 2001 Proceedings 21, pp. 19–40. Springer, Heidelberg (2001)

[CGG24] Chung, K.-M., Goldin, E., Gray, M.: On central primitives for quantum cryptography with classical communication. arXiv preprint arXiv: 2402.17715 [cs.CR] (2024)

[CH14] Chitambar, E., Hsieh, M.-H.: Asymptotic state discrimination and a strict hierarchy in distinguishability norms. J. Math. Phys. **55**(11) (2014)

[CLM+14] Chitambar, E., Leung, D., Mančinska, L., Ozols, M., Winter, A.: Everything you always wanted to know about LOCC (but were afraid to ask). Commun. Math. Phys. **328**, 303–326 (2014)

[CLM23] Chung, K.-M., Lin, Y.-T., Mahmoody, M.: Black-box separations for non-interactive classical commitments in a quantum world. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques, pp. 144–172. Springer, Cham (2023)

[CLMO13] Childs, A.M., Leung, D., Mančinska, L., Ozols, M.: A framework for bounding nonlocality of state discrimination. Commun. Math. Phys. 323, 1121–1153 (2013)

[CLOS02] Canetti, R., Lindell, Y., Ostrovsky, R., Sahai, A.: Universally composable two-party and multi-party secure computation. In: Proceedings of the Thirty-Fourth Annual ACM Symposium on Theory of Computing, pp. 494–503 (2002)

[CM24] Coladangelo, A., Mutreja, S.: On black-box separations of quantum digital signatures from pseudorandom states. arXiv preprint arXiv:2402.08194 (2024)

[Col23] Coladangelo, A.: Quantum trapdoor functions from classical one-way functions. Cryptology ePrint Archive, Paper 2023/282 (2023). https://eprint.iacr.org/2023/282

[DLT02] DiVincenzo, D.P., Leung, D.W., Terhal, B.M.: Quantum data hiding. IEEE Trans. Inf. Theory **48**(3), 580–598 (2002)

[EW02] Eggeling, T., Werner, R.F.: Hiding classical data in multipartite quantum states. Phys. Rev. Lett. **89**(9), 097905 (2002)

[Gea02] Gea-Banacloche, J.: Hiding messages in quantum data. In: J. Math. Phys. **43**(9), 4531–4536 (2002)

[GGM86] Goldreich, O., Goldwasser, S., Micali, S.: How to construct random functions. J. ACM **33**(4), 792–807 (1986)

[GLSV21] Grilo, A.B., Lin, H., Song, F., Vaikuntanathan, V.: Oblivious transfer is in MiniQCrypt. In: Canteaut, A., Standaert, F.-X. (eds.) EUROCRYPT 2021. LNCS, vol. 12697, pp. 531–561. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-77886-6_18

[GS22] Garg, S., Srinivasan, A.: Two-round multiparty secure computation from minimal assumptions. J. ACM **69**(5), 1–30 (2022)

[Har13] Harrow, A.W.: The church of the symmetric subspace. arXiv preprint arXiv:1308.6595 (2013)

[Har23] Harrow, A.W.: Approximate orthogonality of permutation operators, with application to quantum information. Lett. Math. Phys. **114**(1), 1 (2023)

[HBAB19] Halder, S., Banik, M., Agrawal, S., Bandyopadhyay, S.: Strong quantum nonlocality without entanglement. Phys. Rev. Lett. **122**(4), 040403 (2019)

[HLS05] Hayden, P., Leung, D., Smith, G.: Multiparty data hiding of quantum information. Phys. Rev. A **71**(6), 062339 (2005)

[HY20] Hosoyamada, A., Yamakawa, T.: Finding collisions in a quantum world: quantum black-box separation of collision-resistance and one-wayness. In: Advances in Cryptology– ASIACRYPT 2020: 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, 7–11 December 2020, Proceedings, Part I, 26, pp. 3–32. Springer, Cham (2020)

[JLS18] Ji, Z., Liu, Y.-K., Song, F.: Pseudorandom quantum states. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018. LNCS, vol. 10993, pp. 126–152. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-96878-0_5

[Kre21] Kretschmer, W.: Quantum pseudorandomness and classical complexity. In: Hsieh, M.-H. (ed.) 16th Conference on the Theory of Quantum Computation, Communication and Cryptography, TQC 2021, 5–8 July 2021, Virtual Conference. LIPIcs, vol. 197, pp. 2:1–2:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2021). https://doi.org/10.4230/LIPIcs.TQC.2021.2

[LC97] Lo, H.-K., Chau, H.F.: Is quantum bit commitment really possible? Phys. Rev. Lett. **78**(17), 3410 (1997)

[LLLL24] Li, L., Li, Q., Li, X., Liu, Q.: How (not) to build quantum PKE in Minicrypt. arXiv preprint arXiv:2405.20295 (2024)

[May97] Mayers, D.: Unconditionally secure quantum bit commitment is impossible. Phys. Rev. Lett. **78**(17), 3414 (1997)

[MNY23] Morimae, T., Nehoran, B., Yamakawa, T.: Unconditionally secure commitments with quantum auxiliary inputs. arXiv preprint arXiv:2311.18566 [quant-ph] (2023)

[MWW09] Matthews, W., Wehner, S., Winter, A.: Distinguishability of quantum states under restricted families of measurements with an application to quantum data hiding. Commun. Math. Phys. **291**, 813–843 (2009)

[MY21] Morimae, T., Yamakawa, T.: Quantum commitments and signatures without one-way functions. arXiv preprint arXiv:2112.06369 (2021)

[MY23] Morimae, T., Yamakawa, T.: One-wayness in quantum cryptography. arXiv preprint arXiv:2210.03394 [quant-ph] (2023)

[NC10] Nielsen, M.A., Chuang, I.L.: Quantum Computation and Quantum Information: 10th Anniversary Edition. Cambridge University Press (2010). https://doi.org/10.1017/CBO9780511976667

[PNC14] Piani, M., Narasimhachar, V., Calsamiglia, J.: Quantumness of correlations, quantumness of ensembles and quantum data hiding. New J. Phys. **16**(11), 113001 (2014)

[Qia23] Qian, L.: Unconditionally secure quantum commitments with preprocessing. In: Cryptology ePrint Archive (2023)

[RTV04] Reingold, O., Trevisan, L., Vadhan, S.: Notions of reducibility between cryptographic primitives. In: Theory of Cryptography Conference, pp. 1–20. Springer, Heidelberg (2004)

[Yan2] Yan, J.: General properties of quantum bit commitments. In: International Conference on the Theory and Application of Cryptology and Information Security, pp. 628–657. Springer, Cham (2022)

Robust Combiners and Universal Constructions for Quantum Cryptography

Taiga Hiroka[1(✉)], Fuyuki Kitagawa[2,3], Ryo Nishimaki[2,3], and Takashi Yamakawa[1,2,3]

[1] Yukawa Institute for Theoretical Physics, Kyoto University, Kyoto, Japan
taiga.hiroka@yukawa.kyoto-u.ac.jp
[2] NTT Social Informatics Laboratories, Tokyo, Japan
[3] NTT Research Center for Theoretical Quantum Information, Atsugi, Japan

Abstract. A robust combiner combines many candidates for a cryptographic primitive and generates a new candidate for the same primitive. Its correctness and security hold as long as one of the original candidates satisfies correctness and security. A universal construction is a closely related notion to a robust combiner. A universal construction for a primitive is an explicit construction of the primitive that is correct and secure as long as the primitive exists. It is known that a universal construction for a primitive can be constructed from a robust combiner for the primitive in many cases.

Although robust combiners and universal constructions for classical cryptography are widely studied, robust combiners and universal constructions for quantum cryptography have not been explored so far. In this work, we define robust combiners and universal constructions for several quantum cryptographic primitives including one-way state generators, public-key quantum money, quantum bit commitments, and unclonable encryption, and provide constructions of them.

On a different note, it was an open problem how to expand the plaintext length of unclonable encryption. In one of our universal constructions for unclonable encryption, we can expand the plaintext length, which resolves the open problem.

1 Introduction

1.1 Background

The ultimate goal of theoretical cryptography is to construct interesting cryptographic primitives unconditionally. Over the past years, many computational assumptions have been proposed, and many interesting cryptographic primitives have been constructed under the computational assumptions. However, none of the computational assumptions are proven. Indeed, we do not even know how to prove $\mathbf{P} \neq \mathbf{NP}$ while it is a necessary condition to construct interesting classical cryptographic primitives unconditionally. Moreover, given many candidates for a primitive, we cannot often decide which candidate is the most secure one. For example, we can construct public-key encryption (PKE) from decisional Diffie-Hellman (DDH) [DH76, ElG85] or learning with errors (LWE) [Reg05], but currently, we do not know which computational assumption is the weaker assumption.

E. Boyle and M. Mahmoody (Eds.): TCC 2024, LNCS 15365, pp. 126–158, 2025.
https://doi.org/10.1007/978-3-031-78017-2_5

This causes the problem in the following realistic scenario. Suppose we have two candidates for PKE, where one is based on DDH and the other is based on LWE, and we want to decide more secure candidate to use. Unfortunately, in the current knowledge, we cannot decide which candidate is the more secure one.

A robust cryptographic combiner [Her05, HKN+05] was introduced to resolve this issue. Given many candidates for a primitive, a cryptographic combiner combines these candidates and produces a new candidate for the same primitive. The new candidate is correct and secure as long as at least one of the original candidates satisfies correctness and security. For example, a robust PKE combiner takes two candidates for PKE, where one's security relies on DDH and the other's security relies on LWE, and produces a new candidate for PKE. The new candidate is correct and secure as long as the DDH or LWE assumption holds. Robust combiner is a well-studied topic in classical cryptography. In fact, robust combiners for many fundamental classical cryptographic primitives such as one-way functions, public-key encryption, and functional encryption are shown to exist [HKN+05, AJN+16, AJS17, ABJ+19, JMS20].

A closely related notion to a robust combiner is a universal construction [Lev85]. A universal construction for a primitive, say OWFs, is an explicit construction of OWFs that is correct and secure as long as OWFs exist. The adversary must be able to break all OWF candidates to break a universal construction. In this sense, a universal construction for OWFs is the most secure one among all possible OWF candidates. In classical cryptography, universal constructions are well-studied topic and are known to exist for many fundamental primitives. First, the pioneering work by Levin introduces a notion of universal construction and shows how to construct a universal construction for OWFs [Lev85]. After decades, Harnik, Kilian, Naor, Reingold, and Rosen [HKN+05] give a universal construction for PKE and they show how to construct a universal construction for a primitive using a robust combiner for the same primitive. Goldwasser and Kalai cast questions about universal constructions for cryptographic primitives related to obfuscation [GTK16]. The following sequence of works [AJN+16, AJS17, ABJ+19] gives universal constructions for functional encryption under some assumptions, and [JMS20] gives it unconditionally.

Although robust combiners and universal constructions are widely studied topics in classical cryptography, those in the quantum world have not been studied so far, where each party can generate, process, and communicate quantum information. It is well known that, even in the quantum world, information-theoretical security is impossible to achieve for many interesting quantum cryptographic primitives [LC97, May97, Aar18], and currently, many interesting quantum cryptographic primitives are constructed under computational assumptions. For example, public-key quantum money is one of the most interesting quantum cryptographic primitives, and many candidate constructions are proposed relying on computational assumptions [AC12, FGH+12, Kan18, Zha19, KSS22, LMZ23, Zha23b]. However, none of them have been proven so far, and moreover, we cannot even decide which assumptions are the weakest assumptions. This inability leads to the problem that we cannot decide the most secure one to use.

If there exists a robust public-key quantum money combiner, then we can combine them and produce a new candidate for public-key quantum money, which is secure as long as at least one of the original candidates is secure. Therefore, it is natural to ask the following first question:

Is it possible to construct robust combiners for fundamental quantum cryptographic primitives?

On a different note, recent works show the possibility that quantum cryptography exists even if classical cryptography does not. A pseudo-random state generator (PRSG) is a quantum analog of a pseudo-random generator [JLS18], and Kretchmer shows the possibility that PRSGs exist even if **BQP** = **QMA** [Kre21]. Many interesting quantum cryptographic primitives are shown to be constructed from PRSGs [MY22b, MY22a, AQY22, AGQY22, BCQ23]. Among them, one-way state generators (OWSGs) and quantum bit commitments (equivalent to EFI [Yan22, BCQ23]) are considered to be candidates for the necessary assumptions for the existence of quantum cryptography. In the case of classical cryptography, many fundamental primitives have the nice feature of the existence of universal constructions. It is natural to wonder whether quantum cryptographic primitives have universal constructions or not. In fact, some researchers believe that the existence of universal constructions is a nice feature for fundamental cryptographic primitives [Zha23a]. Therefore, we ask the following second question:

Is it possible to construct universal constructions for fundamental quantum cryptographic primitives?

1.2 Our Results

We solve the two questions above affirmatively for several cryptographic primitives. Our contributions to the field are as follows:

1. We formally define robust combiners and universal constructions for many quantum cryptographic primitives including OWSGs, public-key quantum money, quantum bit commitments, and unclonable encryption.
2. We construct a robust combiner and a universal construction for OWSGs without any assumptions. A universal construction is secure as long as there exist OWSGs. In other words, the adversary of a universal construction must be able to break all OWSG candidates. In this sense, our construction for OWSG is the most secure one among all possible OWSG candidates. Before this work, the candidate constructions for OWSGs were based on OWFs, average-case hardness of semi-classical quantum statistical difference [CX22] or random quantum circuits [AQY22, BCQ23] [1].

[1] As discussed in the previous works [AQY22, BCQ23], it is a folklore that a random quantum circuit is PRSGs although there exists no theoretical evidence so far. Since we can construct OWSGs from PRSGs [MY22b, MY22a], we can also construct OWSGs based on random quantum circuits if a random quantum circuit is PRSGs.

3. We construct a robust combiner and a universal construction for public-key quantum money without any assumptions. In particular, in this work, we consider the public-key quantum money mini-scheme introduced in [AC12], which can be generically upgraded into full-fledged public-key quantum money by additionally using digital signatures. A universal construction for a public-key quantum money mini-scheme satisfies security as long as a public-key quantum money mini-scheme exists. In other words, the adversary of a universal construction must be able to break all candidates for a public-key quantum money mini-scheme. In this sense, our construction is the most secure one among all possible public-key quantum money mini-scheme candidates. Before this work, many candidate constructions are proposed [AC12, FGH+12, Kan18, Zha19, KSS22, LMZ23, Zha23b].
4. We construct a robust combiner and a universal construction for quantum bit commitment without any assumptions. Note that our results also imply that we can construct a robust combiner and a universal construction for EFI, oblivious transfer, and multi-party computation, which are equivalent to quantum bit commitments [BCQ23]. In our robust combiner, given n-candidates of quantum bit commitments, we can construct a new quantum bit commitment that satisfies statistical binding and computational hiding at least one of n-candidates satisfies computational hiding and computational binding at the same time. A universal construction for quantum bit commitment is secure as long as there exists a quantum bit commitment. In other words, the adversary for a universal construction must be able to break all candidates for quantum bit commitment. In this sense, our construction for quantum bit commitment is the most secure one among all possible quantum bit commitment candidates. Before this work, candidate constructions of quantum bit commitments were based on OWFs, classical oracle [KQST23], or random quantum circuits [AQY22, BCQ23] [2].
5. We construct robust combiners and universal constructions for various kinds of unclonable encryption as follows:
   - We construct robust combiners for (one-time) unclonable secret-key encryption (SKE) and unclonable public-key encryption (PKE) without any computational assumptions.
   - By using robust combiners, we construct universal constructions for (one-time) unclonable SKE and unclonable PKE without any computational assumptions.

   Although the previous work [AKL+22] gives a construction of one-time unclonable SKE with unclonable IND-CPA security in the quantum random oracle model (QROM), it was an open problem to construct it in the standard model. Our universal constructions for (one-time) unclonable SKE (resp. PKE) is the first construction of (one-time) unclonable SKE (resp. PKE) that

[2] It is a folklore that a random quantum circuit is PRSGs although there exists no theoretical evidence so far. Since we can construct quantum bit commitments from PRSGs [MY22b, AQY22], we can also construct quantum bit commitments based on random quantum circuits if a random quantum circuit is PRSGs.

achieves unclonable IND-CPA security in the standard model, where the security relies on the existence of (one-time) unclonable SKE (resp. PKE) with unclonable IND-CPA security.

6. We give another construction of universal construction for one-time unclonable SKE by additionally using the decomposable quantum randomized encoding, which can be instantiated by OWFs [BY22]. Although this construction additionally uses decomposable quantum randomized encoding, it has the following nice three properties that the universal construction via a robust combiner does not have:
   - It was an open problem whether unclonable encryption with single-bit plaintexts implies unclonable encryption with multi-bit plaintexts because standard transformation via bit-wise encryption does not work as pointed out in [AKL+22]. In our universal construction, we can expand the plaintext length of one-time unclonable SKE by additionally using decomposable quantum randomized encoding. This resolves the open problem left by [AKL+22]. Note that this result implies that reusable unclonable SKE and unclonable PKE can expand plaintext length without any additional assumptions because reusable unclonable SKE and unclonable PKE imply decomposable quantum randomized encoding.
   - A universal construction via a robust combiner needs to emulate all possible algorithms, and thus a huge constant is included in the running time. Therefore, it may not be executed in a meaningful amount of time if we want reasonable concrete security. On the other hand, universal construction via decomposable quantum randomized encoding does not emulate all possible algorithms and thus avoids the "galactic inefficiency" tied to such approaches.
   - In a universal construction via a robust combiner, the security relies on the existence of one-time unclonable SKE scheme $\Sigma = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$, where $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ are uniform QPT algorithms. On the other hand, in a universal construction via decomposable quantum randomized encoding, the security still holds even if the underlying one-time unclonable SKE $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ are non-uniform algorithms.

1.3 More on Related Work

Fundamental Quantum Cryptographic Primitives. Ji, Liu, and Song [JLS18] introduce a notion of PRSGs, and show that it can be constructed from OWFs. Morimae and Yamakawa [MY22b] introduce the notion of OWSGs, and show how to construct them from PRSGs. In the first definition of OWSGs, the output quantum states are restricted to pure states, and its definition is generalized to mixed states by [MY22a]. In this work, we focus on the mixed-state version.

Bennett and Brassard [BB84] initiate the study of quantum bit commitment. Unfortunately, it turns out that statistically secure quantum bit commitments are impossible to achieve [LC97, May97]. Therefore, later works study a quantum bit commitment with computational security [DMS00, CLS01, Yan22, MY22b, MY22a, AQY22, AGQY22, BCQ23, HMY23]. It was shown that quantum

bit commitments can be constructed from PRSGs by [MY22b, AQY22], and that quantum bit commitments are equivalent to EFI, oblivious transfer, and multi-party computation [GLSV21, BCKM21, Yan22, BCQ23].

Recently, Khurana and Tomer [KT23] showed that quantum bit commitments can be constructed from OWSGs with pure state. Although their main result is not a combiner for quantum bit commitment, they construct some sort of a combiner for quantum bit commitments as an intermediate tool for achieving their result. In their construction, they construct a uniform quantum bit commitment from a non-uniform one. At this step, they combine quantum bit commitments in the following sense. In their construction, they combine $(n+1)$-quantum bit commitments and generate a new quantum bit commitment. Its hiding and binding property holds as long as one of the original candidates satisfies hiding and binding at the same time and other n candidates also satisfy either hiding or binding. Compared to their technique, our robust combiner does not need to assume other n candidates satisfy hiding or binding. Therefore, our robust combiner can be applied in a more general setting than their technique. Though our construction partially shares a similarity with theirs, we rely on additional ideas to deal with candidate schemes that do not satisfy either binding or hiding.

Unclonable Encryption. Broadbent and Lord [BL20] introduced a notion of unclonable encryption. They considered two security definitions for unclonable encryption. One is one-wayness against cloning attacks and they achieve information-theoretic one-wayness by using BB84 states. The other is indistinguishability against cloning attacks (indistinguishable-secure unclonable encryption). However, they did not achieve it. They constructed indistinguishable-secure unclonable encryption only in a very restricted model by using PRFs. Ananth, Kaleoglu, Li, Liu, and Zhandry [AKL+22] proposed the first indistinguishable-secure unclonable encryption in the QROM. Ananth and Kaleoglu [AK21] construct unclonable PKE from unclonable encryption and PKE with "classical" ciphertexts. Note that it is unclear how to apply their technique for PKE with quantum ciphertexts. The technique of [HMNY21] can be used to construct unclonable PKE from unclonable encryption and PKE with quantum ciphertexts, which we use in this work.

Combiner for Classical Cryptography. It is known that robust combiners are known to exist for many fundamental classical cryptographic primitives. Oblivious transfer (OT) is an example of exceptions. It is an open problem how to construct a robust combiner for classical OT and some black-box impossibilities are known [HKN+05]. Interestingly, our result implies that a robust combiner for quantum OT exists although a robust combiner for classical OT is still an open problem.

2 Technical Overview

First of all, let us recall the definition of robust combiner. A robust combiner for a primitive P is a deterministic classical polynomial-time Turing machine

$\mathsf{RobComb}.\mathcal{M}_P$ that takes as input n-candidates $\{\Sigma[i]\}_{i\in[n]}$ for P, and produces a new candidate Σ for P. Σ is correct and secure as long as at least one of the candidates $\{\Sigma[i]\}_{i\in[n]}$ for P is correct and secure. Here, the point is that $\{\Sigma[i]\}_{i\in[n]}$ are not promised to satisfy even correctness other than one of them. In the following, we will explain the case where only two candidates $\Sigma[1]$ and $\Sigma[2]$ are given for simplicity. Remark that the same argument goes through in the general case, where n candidates $\{\Sigma[i]\}_{i\in[n]}$ are given.

2.1 Robust Combiner for One-Way State Generators and Public- Key Quantum Money

In this section, we explain a robust combiner for OWSGs. A robust combiner for public-key quantum money can be constructed by partially using the technique by [HKN+05].

Definition of One-Way State Generators. OWSG is a quantum generalization of OWFs and consists of a tuple of quantum polynomial-time algorithms $\Sigma_{\mathsf{OWSG}} := (\mathsf{KeyGen}, \mathsf{StateGen}, \mathsf{Vrfy})$. The KeyGen algorithm takes as input a security parameter 1^λ, and generates a classical key k, the $\mathsf{StateGen}$ algorithm takes as input a classical key k and outputs a quantum state ψ_k, and the Vrfy algorithm takes as input a classical key k and a quantum state ψ_k and outputs 1 indicating acceptance or 0 indicating rejection. We require that OWSG Σ satisfies correctness and security. The correctness guarantees that $\mathsf{Vrfy}(k, \psi_k)$ outputs 1 indicating acceptance with overwhelming probability, where $k \leftarrow \mathsf{KeyGen}(1^\lambda)$ and $\psi_k \leftarrow \mathsf{StateGen}(k)$. The security guarantees that no QPT adversaries given polynomially many copies of ψ_k cannot generate k^* such that $1 \leftarrow \mathsf{Vrfy}(k^*, \psi_k)$, where $k \leftarrow \mathsf{KeyGen}(1^\lambda)$ and $\psi_k \leftarrow \mathsf{StateGen}(k)$.

Robust Combiner. First, we consider the simpler case, where given OWSG candidates $\Sigma_{\mathsf{OWSG}}[1] = (\mathsf{KeyGen}[1], \mathsf{StateGen}[1], \mathsf{Vrfy}[1])$ and $\Sigma_{\mathsf{OWSG}}[2] = (\mathsf{KeyGen}[2], \mathsf{StateGen}[2], \mathsf{Vrfy}[2])$ are promised to satisfy at least correctness. In this case, we can construct a combiner for OWSGs in the same way as OWFs. Namely, a combined protocol $\mathsf{Comb}.\Sigma_{\mathsf{OWSG}} = (\mathsf{KeyGen}, \mathsf{StateGen}, \mathsf{Vrfy})$ simply runs $\Sigma[1]$ and $\Sigma[2]$ in parallel.

Does the same strategy work for the general setting, where original candidates are not promised to satisfy correctness? Unfortunately, the simple parallel protocol works only when both $\Sigma_{\mathsf{OWSG}}[1]$ and $\Sigma_{\mathsf{OWSG}}[2]$ satisfy correctness because $\mathsf{Comb}.\Sigma_{\mathsf{OWSG}}$ does not satisfy correctness otherwise. We observe that given an OWSG candidate Σ_{OWSG}, we can construct Σ^*_{OWSG} with the following properties:

- Σ^*_{OWSG} satisfies correctness regardless of Σ_{OWSG}.
- Σ^*_{OWSG} satisfies security as long as Σ_{OWSG} satisfies correctness and security.

Once we have obtained such a transformation, we can construct a robust OWSG combiner $\mathsf{RobComb}.\mathcal{M}_{\mathsf{OWSG}}$ as follows. Given two OWSGs candidates $\Sigma_{\mathsf{OWSG}}[1]$ and $\Sigma_{\mathsf{OWSG}}[2]$, our robust OWSG combiner $\mathsf{RobComb}.\mathcal{M}_{\mathsf{OWSG}}$ first

transforms them into $\Sigma_{\mathsf{OWSG}}[1]^*$ and $\Sigma_{\mathsf{OWSG}}[2]^*$, respectively, and then outputs $\mathsf{Comb}.\Sigma_{\mathsf{OWSG}}$ which runs $\Sigma_{\mathsf{OWSG}}[1]^*$ and $\Sigma_{\mathsf{OWSG}}[2]^*$ in parallel. $\mathsf{Comb}.\Sigma_{\mathsf{OWSG}}$ satisfies correctness because $\Sigma_{\mathsf{OWSG}}[1]^*$ and $\Sigma_{\mathsf{OWSG}}[2]^*$ satisfies correctness no matter what $\Sigma_{\mathsf{OWSG}}[1]$ and $\Sigma_{\mathsf{OWSG}}[2]$ are. $\mathsf{Comb}.\Sigma_{\mathsf{OWSG}}$ satisfies security as long as either $\Sigma_{\mathsf{OWSG}}[1]$ or $\Sigma_{\mathsf{OWSG}}[2]$ satisfy correctness and security because either $\Sigma_{\mathsf{OWSG}}[1]^*$ or $\Sigma_{\mathsf{OWSG}}[2]^*$ satisfies security as long as either $\Sigma_{\mathsf{OWSG}}[1]$ or $\Sigma_{\mathsf{OWSG}}[2]$ satisfies correctness and security.

Transform Incorrect Candidate into Correct One. Now, we consider how to obtain such a transformation. In the previous work [HKN+05], it was shown that we can transform PKE Σ_{PKE} into Σ^*_{PKE} that satisfies correctness regardless of Σ_{PKE} and satisfies security as long as Σ_{PKE} satisfies correctness and security. In the same way as [HKN+05], we can obtain such transformation for OWSGs. However, in this work, we take a different approach because the technique by [HKN+05] does not work for unclonable encryption[3].

First, we observe that without loss of generality, $\mathsf{Vrfy}(k,\psi)$ can be considered working as follows: It appends $|0\rangle\langle 0|$ to ψ, applies U_k to $\psi\otimes|0\rangle\langle 0|$, measures the first qubit of $U_k(\psi\otimes|0\rangle\langle 0|)U_k^\dagger$, and outputs the measurement outcome. Now, we describe $\Sigma^*_{\mathsf{OWSG}}=(\mathsf{KeyGen}^*,\mathsf{StateGen}^*,\mathsf{Vrfy}^*)$. KeyGen^* is the same as the original KeyGen. $\mathsf{StateGen}^*(k)$ first runs $\psi_k\leftarrow\mathsf{StateGen}(k)$, then measures the first qubit of $U_k(\psi_k\otimes|0\rangle\langle 0|)U_k^\dagger$ in the computational basis, and obtains b. If $b=1$, $\mathsf{StateGen}^*(k)$ rewinds its register and outputs the register as ψ_k^*. Otherwise, output $\psi_k^*=\bot$, where $\bot$ is a special symbol. $\mathsf{Vrfy}^*(k,\psi)$ first checks the form of ψ. If $\psi=\bot$, $\mathsf{Vrfy}^*(k,\psi)$ outputs 1. Otherwise, $\mathsf{Vrfy}^*(k,\psi)$ applies U_k to ψ, then measures the first qubit of $U_k\psi U_k^\dagger$, and finally outputs the measurement outcome. We can see that Σ^* satisfies correctness. If $\mathsf{StateGen}^*(k)$ outputs $\psi_k^*=\bot$, then Vrfy^* always outputs 1. On the other hand, if $\psi^*\neq\bot$, then $\mathsf{StateGen}^*(k)$ outputs ψ_k^* with the form $U_k^\dagger(|1\rangle\langle 1|\otimes\rho)U_k$ for some quantum state ρ. Therefore, $\mathsf{Vrfy}^*(k,\psi_k^*)$ outputs 1 since $U_k\psi_k^*U_k^\dagger=|1\rangle\langle 1|\otimes\rho$. Moreover, we can see that Σ^* satisfies security as long as Σ satisfies correctness and security. As long as Σ satisfies correctness, if we measure the first qubits of $U_k(\psi_k\otimes|0\rangle\langle 0|)U_k^\dagger$ in the computational basis, then the measurement result is 1 with overwhelming probability, where $k\leftarrow\mathsf{KeyGen}(1^\lambda)$ and $\psi_k\leftarrow\mathsf{StateGen}(k)$. This indicates that the measurement does not disturb the quantum state $U_k(\psi_k\otimes|0\rangle\langle 0|)U_k^\dagger$ from gentle measurement lemma. Therefore, ψ_k^* is statistically close to $\psi_k\otimes|0\rangle\langle 0|$ as long as Σ satisfies correctness. In particular, this implies that we can reduce the security of Σ^* to that of Σ as long as Σ satisfies correctness.

[3] The technique we introduce here cannot be applied to public-key quantum money. For public-key quantum money, we apply the technique introduced by [HKN+05] in order to transform an incorrect candidate into a correct one. The idea of transformation is first checking the correctness of a public-key quantum money candidate $\Sigma=(\mathsf{Mint},\mathsf{Vrfy})$. If the candidate Σ satisfies the correctness, then we amplify the correctness by parallel repetition. Otherwise, we use the scheme $\Sigma^*=(\mathsf{Mint}^*,\mathsf{Vrfy}^*)$, where Vrfy^* algorithm always outputs $\top$. For details, please see the full version.

2.2 Robust Combiner for Unclonable Encryption

In this section, we explain how to obtain a robust combiner for unclonable SKE. As a corollary, we can obtain a robust combiner for unclonable PKE. This is because we can construct unclonable PKE from unclonable SKE and PKE with quantum ciphertexts [HMNY21, AK21], and a robust combiner for PKE with quantum ciphertexts can be constructed in the same way as the classical ciphertexts case [HKN+05].

Definition of Unclonable SKE. First of all, we explain the definition of unclonable SKE. Unclonable SKE $\Sigma_{\mathsf{unclone}}$ is the same as standard SKE Σ_{SKE} except that the ciphertext of unclonable SKE is a quantum state and it satisfies unclonable IND-CPA security in addition to standard IND-CPA security. In unclonable IND-CPA security, the cloning adversary $\mathcal{A}$ with oracle $\mathsf{Enc}(\mathsf{sk}, \cdot)$ first sends the challenge plaintext (m_0, m_1), then receives a ciphertext CT_b, where $\mathsf{CT}_b \leftarrow \mathsf{Enc}(\mathsf{sk}, m_b)$, and finally generates a quantum state $\rho_{\mathcal{B},\mathcal{C}}$ over the $\mathcal{B}$ and $\mathcal{C}$ registers. The adversary $\mathcal{B}$ (resp. $\mathcal{C}$) receives the $\mathcal{B}$ register (resp. the $\mathcal{C}$ register) and the secret-key sk, and outputs $b_{\mathcal{B}}$ (resp. $b_{\mathcal{C}}$) which is a guess of b. The unclonable IND-CPA security guarantees that for any QPT adversaries $(\mathcal{A}, \mathcal{B}, \mathcal{C})$, we have

$$\Pr[b = b_{\mathcal{B}} = b_{\mathcal{C}}] \leq \frac{1}{2} + \mathsf{negl}(\lambda).$$

Robust Combiner. First, we consider the simpler case, where given candidates $\Sigma_{\mathsf{unclone}}[1] = (\mathsf{KeyGen}[1], \mathsf{Enc}[1], \mathsf{Dec}[1])$ and $\Sigma_{\mathsf{unclone}}[2] = (\mathsf{KeyGen}[2], \mathsf{Enc}[2], \mathsf{Dec}[2])$ are promised to satisfy at least correctness. In that case, a combined unclonable SKE scheme $\mathsf{Comb}.\Sigma_{\mathsf{unclone}} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ simply runs $\Sigma_{\mathsf{unclone}}[1]$ and $\Sigma_{\mathsf{unclone}}[2]$ by using X-OR secret sharing. In other words, for encrypting bit b, $\mathsf{Comb}.\Sigma_{\mathsf{unclone}}$ first samples $r[1]$ and $r[2]$ such that $r[1] + r[2] = b$, and encrypts $r[1]$ by using $\Sigma_{\mathsf{unclone}}[1]$ and $r[2]$ by using $\Sigma_{\mathsf{unclone}}[2]$. Clearly, $\mathsf{Comb}.\Sigma_{\mathsf{unclone}}$ satisfies correctness and security as long as both $\Sigma_{\mathsf{unclone}}[1]$ and $\Sigma_{\mathsf{unclone}}[2]$ satisfy correctness and either $\Sigma_{\mathsf{unclone}}[1]$ or $\Sigma_{\mathsf{unclone}}[2]$ satisfies security.

Does the same strategy work for the general setting, where original candidates are not promised to satisfy even correctness? Unfortunately, the simple X-OR protocol above works only when both $\Sigma_{\mathsf{unclone}}[1]$ and $\Sigma_{\mathsf{unclone}}[2]$ satisfy correctness because $\mathsf{Comb}.\Sigma_{\mathsf{unclone}}$ does not satisfy correctness otherwise. Our key observation is that given a candidate of unclonable SKE $\Sigma_{\mathsf{unclone}}$ we can construct a new candidate $\Sigma^*_{\mathsf{unclone}}$ with the following properties:

- $\Sigma^*_{\mathsf{unclone}}$ satisfies correctness regardless of $\Sigma_{\mathsf{unclone}}$.
- $\Sigma^*_{\mathsf{unclone}}$ satisfies security as long as Σ satisfies correctness and security.

Once we have obtained such a transformation, we can construct a robust combiner for unclonable SKE as follows. Given two unclonable SKE candidates $\Sigma_{\mathsf{unclone}}[1]$ and $\Sigma_{\mathsf{unclone}}[2]$, a robust combiner for unclonable SKE first transforms $\Sigma_{\mathsf{unclone}}[1]$ and $\Sigma_{\mathsf{unclone}}[2]$ into $\Sigma_{\mathsf{unclone}}[1]^*$ and $\Sigma_{\mathsf{unclone}}[2]^*$, respectively,

and then outputs $\mathsf{Comb}.\Sigma_{\mathsf{unclone}}$ which runs $\Sigma_{\mathsf{unclone}}[1]^*$ and $\Sigma_{\mathsf{unclone}}[2]^*$ by using X-OR secret sharing. $\mathsf{Comb}.\Sigma_{\mathsf{unclone}}$ satisfies correctness because $\Sigma_{\mathsf{unclone}}[1]^*$ and $\Sigma_{\mathsf{unclone}}[2]^*$ satisfy correctness no matter what $\Sigma_{\mathsf{unclone}}[1]$ and $\Sigma_{\mathsf{unclone}}[2]$ are. Moreover, $\mathsf{Comb}.\Sigma_{\mathsf{unclone}}$ satisfies security as long as either $\Sigma_{\mathsf{unclone}}[1]$ or $\Sigma_{\mathsf{unclone}}[2]$ satisfies correctness and security. This is because either $\Sigma^*_{\mathsf{unclone}}[1]$ or $\Sigma_{\mathsf{unclone}}[2]^*$ satisfies security as long as either $\Sigma_{\mathsf{unclone}}[1]$ or $\Sigma_{\mathsf{unclone}}[2]$ satisfies correctness and security.

Transform Incorrect Candidate into Correct One. Now, we consider how to obtain such a transformation. It is known that we can obtain such a transformation for PKE [HKN+05]. In their technique, they use parallel repetition to amplify correctness. We emphasize that we cannot apply their technique for unclonable encryption because correctness amplification via parallel repetition does not work for unclonable encryption. Therefore, we take a different approach, whose idea is the same as OWSGs. Without loss of generality, we can assume that $\mathsf{Dec}(\mathsf{sk}, \mathsf{CT})$ first appends $|0\rangle\langle 0|$ to CT, applies U_{sk} to $\mathsf{CT} \otimes |0\rangle\langle 0|$, measures the first $|m|$-bit of $U_{\mathsf{sk}}(\mathsf{CT} \otimes |0\rangle\langle 0|)U_{\mathsf{sk}}^\dagger$, and outputs the measurement outcome. Now, we describe $\Sigma^*_{\mathsf{unclone}} = (\mathsf{KeyGen}^*, \mathsf{Enc}^*, \mathsf{Dec}^*)$. KeyGen^* is the same as the original KeyGen. $\mathsf{Enc}^*(\mathsf{sk}, m)$ first runs $\mathsf{CT} \leftarrow \mathsf{Enc}(\mathsf{sk}, m)$, then measures the first $|m|$-bit of $U_{\mathsf{sk}}(\mathsf{CT} \otimes |0\rangle\langle 0|)U_{\mathsf{sk}}^\dagger$ in the computational basis, obtains m^*, and checks whether $m = m^*$. If $m = m^*$, $\mathsf{Enc}^*(\mathsf{sk}, \mathsf{CT})$ rewinds its register and outputs the register as the quantum ciphertext CT^*. Otherwise, output $\mathsf{CT}^* = (\bot, m)$, where $\bot$ is a special symbol. $\mathsf{Dec}^*(\mathsf{sk}, \mathsf{CT}^*)$ first checks the form of CT^*, and outputs m if CT^* is of the form $(\bot, m)$. Otherwise, $\mathsf{Dec}^*(\mathsf{sk}, \mathsf{CT}^*)$ applies U_{sk} to CT^*, and outputs the measurement outcome of first $|m|$-qubits of $U_{\mathsf{sk}}\mathsf{CT}^*U_{\mathsf{sk}}^\dagger$. Clearly, the new construction $\Sigma^*_{\mathsf{unclone}}$ satisfies correctness in the same reason as OWSG. Furthermore, $\Sigma^*_{\mathsf{unclone}}$ satisfies security as long as $\Sigma_{\mathsf{unclone}}$ satisfies correctness and security. This is because CT^* is statistically close to $\mathsf{CT} \otimes |0\rangle\langle 0|$ as long as $\Sigma_{\mathsf{unclone}}$ satisfies correctness, and thus we can reduce the security of $\Sigma^*_{\mathsf{unclone}}$ to that of $\Sigma_{\mathsf{unclone}}$.

2.3 Robust Combiner for Quantum Bit Commitment

Definition of Quantum Bit Commitment. In the following, we consider a robust combiner for quantum bit commitment. In this work, we consider a canonical quantum bit commitment. Any quantum bit commitment can be written in the following canonical form [Yan22]. A canonical quantum bit commitment scheme is a pair of unitaries (Q_0, Q_1) acting on the registers $\mathbf{C}$ called the commitment register and $\mathbf{R}$ called the reveal register, and works as follows.

Commit Phase: A sender runs $Q_b|0\rangle_{\mathbf{C},\mathbf{R}}$ and sends the $\mathbf{C}$ to a receiver for committing a bit $b \in \{0, 1\}$.

Reveal Phase: For revealing the committed bit b, the sender sends b and the $\mathbf{R}$ register to the receiver. The receiver applies $Q_b^\dagger$ to the $\mathbf{C}$ and $\mathbf{R}$ register and measures both registers in the computational basis. The receiver accepts if the measurement outcomes are all 0, and rejects otherwise.

We require that a canonical quantum bit commitment satisfies hiding and binding. The computational (resp. statistical) hiding requires that no quan-

tum polynomial-time (resp. unbounded) adversaries distinguish $Q_0|0\rangle_{\mathbf{C},\mathbf{R}}$ from $Q_1|0\rangle_{\mathbf{C},\mathbf{R}}$ without touching the $\mathbf{R}$ register with non-negligible probability.

The binding requires that no adversaries can map an honestly generated quantum bit commitment of 0 (i.e. $Q_0|0\rangle_{\mathbf{C},\mathbf{R}}$) to that of 1 (i.e. $Q_1|0\rangle_{\mathbf{C},\mathbf{R}}$) without touching $\mathbf{C}$ registers. More formally, computational (resp. statistical) binding requires that for any quantum polynomial-time (resp. unbounded) unitary $U_{\mathbf{R},\mathbf{Z}}$ acting on the $\mathbf{R}$ and $\mathbf{Z}$ register and any quantum state $|\tau\rangle_{\mathbf{Z}}$ on $\mathbf{Z}$ register, we have

$$\left\|(Q_1|0\rangle\langle 0|Q_1^\dagger)_{\mathbf{C},\mathbf{R}}(I_{\mathbf{C}}\otimes U_{\mathbf{R},\mathbf{Z}})(Q_0|0\rangle_{\mathbf{C},\mathbf{R}}|\tau\rangle_{\mathbf{Z}})\right\| \leq \mathsf{negl}(\lambda).$$

It was shown that we can change the flavor of quantum bit commitment [Yan22, HMY23]. More formally, if we have a canonical quantum bit commitment (Q_0, Q_1) that satisfies X-hiding and Y-binding, then we can construct a canonical quantum bit commitment $(\widetilde{Q_0}, \widetilde{Q_1})$ that satisfies X-binding and Y-hiding for $X, Y \in \{\text{statistical, computational}\}$.

Robust Combiner. First, let us clarify our final goal. Given two candidates of canonical quantum bit commitments $(Q_0[1], Q_1[1])$ and $(Q_0[2], Q_1[2])$, our robust combiner $\mathsf{RobComb}.\mathcal{M}_{\mathsf{Commit}}$ generates a new candidate $(\mathsf{Comb}.Q_0, \mathsf{Comb}.Q_1)$ that satisfies hiding and binding as long as either $(Q_0[1], Q_1[1])$ or $(Q_0[2], Q_1[2])$ satisfies hiding and binding. More formally, our robust combiner $\mathsf{RobComb}.\mathcal{M}_{\mathsf{Commit}}$ outputs $(\mathsf{Comb}.Q_0, \mathsf{Comb}.Q_1)$ with the following properties:

- $(\mathsf{Comb}.Q_0, \mathsf{Comb}.Q_1)$ satisfies statistical binding regardless of $(Q_0[1], Q_1[1])$ and $(Q_0[2], Q_1[2])$.
- $(\mathsf{Comb}.Q_0, \mathsf{Comb}.Q_1)$ satisfies computational hiding as long as either $(Q_0[1], Q_1[1])$ or $(Q_0[2], Q_1[2])$ satisfies computational hiding and computational binding.

To achieve this final goal, let us consider the following simpler goal first, where both candidates $(Q_0[1], Q_1[1])$ and $(Q_0[2], Q_1[2])$ satisfy at least statistical binding. More formally, given candidates $(Q_0[1], Q_1[1])$ and $(Q_0[2], Q_1[2])$, we consider constructing a new candidate $(\mathsf{Comb}.Q_0, \mathsf{Comb}.Q_1)$ with the following properties:

- $(\mathsf{Comb}.Q_0, \mathsf{Comb}.Q_1)$ satisfies statistical binding as long as both $(Q_0[1], Q_1[1])$ and $(Q_0[2], Q_1[2])$ satisfies statistical binding.
- $(\mathsf{Comb}.Q_0, \mathsf{Comb}.Q_1)$ satisfies computational hiding as long as either $(Q_0[1], Q_1[1])$ or $(Q_0[2], Q_1[2])$ satisfies computational hiding.

We can construct such $(\mathsf{Comb}.Q_0, \mathsf{Comb}.Q_1)$ by simply using X-OR secret sharing. More formally, for $b \in \{0, 1\}$, $\mathsf{Comb}.Q_b$ first samples $r[1]$ and $r[2]$ conditioned on $r[1] + r[2] = b$, and then commits $r[1]$ by using $(Q_0[1], Q_1[1])$ and commits $r[2]$ by using $(Q_0[2], Q_1[2])$. Our construction satisfies statistical binding as long as both $(Q_0[1], Q_1[1])$ and $(Q_0[2], Q_1[2])$ satisfy statistical binding. The intuitive reason is that the adversary of $(\mathsf{Comb}.Q_0, \mathsf{Comb}.Q_1)$ needs to change $r[1]$ or $r[2]$ after sending the commitment register to break

binding of $(\mathsf{Comb}.Q_0, \mathsf{Comb}.Q_1)$, but the adversary cannot do this because both $(Q_0[1], Q_1[1])$ and $(Q_0[2], Q_1[2])$ satisfy statistical binding. Furthermore, $(\mathsf{Comb}.Q_0, \mathsf{Comb}.Q_1)$ satisfies computational hiding as long as either $(Q_0[1], Q_1[1])$ or $(Q_0[2], Q_1[2])$ satisfies computational hiding. The intuitive reason is that the adversary of $(\mathsf{Comb}.Q_0, \mathsf{Comb}.Q_1)$ needs to obtain both $r[1]$ and $r[2]$ from the commitment register of $(Q_0[1], Q_1[1])$ and $(Q_0[2], Q_1[2])$, but the adversary cannot do this because either $(Q_0[1], Q_1[1])$ and $(Q_0[2], Q_1[2])$ satisfies computational hiding.

Does the same strategy work for a robust quantum bit commitment combiner $\mathsf{RobComb}.\mathcal{M}_{\mathsf{Commit}}$? Unfortunately, the simple X-OR protocol above works only when both $(Q_0[1], Q_1[1])$ and $(Q_0[2], Q_1[2])$ satisfy statistical binding because $(\mathsf{Comb}.Q_0, \mathsf{Comb}.Q_1)$ does not satisfy statistical binding otherwise. Our key observation is that, given a candidate of canonical quantum bit commitment (Q_0, Q_1), we can construct a new candidate (Q_0^*, Q_1^*) that satisfies at least statistical binding regardless of (Q_0, Q_1). More formally, we can construct (Q_0^*, Q_1^*) with the following properties:

- (Q_0^*, Q_1^*) satisfies statistical binding regardless of (Q_0, Q_1).
- (Q_0^*, Q_1^*) satisfies computational hiding if (Q_0, Q_1) satisfies computational hiding and computational binding.

Once we have obtained such a transformation, we can construct a robust quantum bit commitment combiner $\mathsf{RobComb}.\mathcal{M}_{\mathsf{Commit}}$. Given two candidates of canonical quantum bit commitment $(Q_0[1], Q_1[1])$ and $(Q_0[2], Q_1[2])$, $\mathsf{RobComb}.\mathcal{M}_{\mathsf{Commit}}$ first transforms $(Q_0[1], Q_1[1])$ and $(Q_0[2], Q_1[2])$ into $(Q_0[1]^*, Q_1[1]^*)$ and $(Q_0[2]^*, Q_1[2]^*)$, respectively and then outputs $(\mathsf{Comb}.Q_0, \mathsf{Comb}.Q_1)$, which runs $(Q_0[1]^*, Q_1[1]^*)$ and $(Q_0[2]^*, Q_1[2]^*)$ by using X-OR secret sharing. Clearly, $(\mathsf{Comb}.Q_0, \mathsf{Comb}.Q_1)$ satisfies statistical binding. Moreover, $(\mathsf{Comb}.Q_0, \mathsf{Comb}.Q_1)$ satisfies computational hiding as long as either $(Q_0[1], Q_1[1])$ or $(Q_0[2], Q_1[2])$ satisfies computational hiding and computational binding.

Transform Candidate Without Statistical Binding into One with Statistical Binding. Now, we consider how to obtain such a transformation. Our first observation is that either (Q_0, Q_1) or $(\widetilde{Q_0}, \widetilde{Q_1})$, which is the flavor conversion of (Q_0, Q_1) obtained by [HMY23], satisfies statistical binding in a possibly weak sense. To see this let us denote $\rho_b := \mathrm{Tr}_{\mathbf{R}}(Q_b|0\rangle_{\mathbf{C},\mathbf{R}})$. Then, there exists some constant f such that

$$F(\rho_0, \rho_1) = f,$$

where $F(\rho_0, \rho_1)$ is the fidelity between ρ_0 and ρ_1. If f is small, then (Q_0, Q_1) satisfies statistical binding in a possibly weak sense from Uhlmann's theorem. On the other hand, if f is large, then (Q_0, Q_1) does not satisfy statistical binding, but $(\widetilde{Q_0}, \widetilde{Q_1})$ satisfies statistical binding instead. This is because if f is large, then (Q_0, Q_1) satisfies statistical hiding, and thus $(\widetilde{Q_0}, \widetilde{Q_1})$ satisfies statistical binding. Therefore, either (Q_0, Q_1) or $(\widetilde{Q_0}, \widetilde{Q_1})$ satisfies statistical binding in a possibly weak sense regardless of (Q_0, Q_1). Furthermore, we observe that such

a possibly weak binding property can be amplified to a strong one by parallel repetition.

Based on these observations, we construct our transformation. Given a candidate of canonical quantum bit commitment (Q_0, Q_1), our transformation outputs a new candidate (Q_0^*, Q_1^*) working as follows.

- If we write $\mathbf{C}$ and $\mathbf{R}$ to mean the commitment register and the reveal register of (Q_0, Q_1), and write $\widetilde{\mathbf{C}}$ and $\widetilde{\mathbf{R}}$ to mean the commitment and the reveal register of $(\widetilde{Q_0}, \widetilde{Q_1})$, then the commitment register $\mathbf{C}^*$ of (Q_0^*, Q_1^*) is $(\mathbf{C}^{\otimes\lambda}, \widetilde{\mathbf{C}}^{\otimes\lambda})$, and the reveal register $\mathbf{R}^*$ of (Q_0^*, Q_1^*) is $(\mathbf{R}^{\otimes\lambda}, \widetilde{\mathbf{R}}^{\otimes\lambda})$.
- For $b \in \{0,1\}$, Q_b^* works as follows:

$$Q_b^* \coloneqq (Q_b \otimes \widetilde{Q_b})^{\otimes\lambda}.$$

Note that we have

$$Q_b^*|0\rangle_{\mathbf{C}^*,\mathbf{R}^*} = (Q_b|0\rangle_{\mathbf{C},\mathbf{R}})^{\otimes\lambda} \otimes (\widetilde{Q_b}|0\rangle_{\widetilde{\mathbf{C}},\widetilde{\mathbf{R}}})^{\otimes\lambda}.$$

We can see that (Q_0^*, Q_1^*) satisfies statistical binding regardless of (Q_0, Q_1). If we write $\rho_b \coloneqq \mathrm{Tr}_{\mathbf{R}}(Q_b|0\rangle_{\mathbf{C},\mathbf{R}})$, there exists some constant $0 \le f \le 1$ such that

$$F(\rho_0, \rho_1) = f.$$

If we write $\widetilde{\rho_b} \coloneqq \mathrm{Tr}_{\widetilde{\mathbf{R}}}(\widetilde{Q_b}|0\rangle_{\widetilde{\mathbf{C}},\widetilde{\mathbf{R}}})$, then we can show that

$$F(\widetilde{\rho_0}, \widetilde{\rho_1}) \le (1-f)^{1/2}$$

by using the technique by [HMY23]. Therefore, if we write $\rho_b^* \coloneqq \mathrm{Tr}_{\mathbf{R}^*}(Q_b^*|0\rangle_{\mathbf{C}^*,\mathbf{R}^*})$, we have

$$F(\rho_0^*, \rho_1^*) = F((\rho_0 \otimes \widetilde{\rho_0})^{\otimes\lambda}, (\rho_1 \otimes \widetilde{\rho_1})^{\otimes\lambda}) \le F(\rho_0, \rho_1)^\lambda F(\widetilde{\rho_0}, \widetilde{\rho_1})^\lambda \le f^\lambda (1-f)^{\lambda/2} \le 2^{-\lambda/2}.$$

This implies that (Q_0^*, Q_1^*) satisfies statistical binding regardless of (Q_0, Q_1) from Uhlmann's Theorem.

Moreover, we can see that (Q_0^*, Q_1^*) satisfies computational hiding as long as (Q_0, Q_1) satisfies computational hiding and computational binding. The hiding QPT adversary of (Q_0^*, Q_1^*) needs to obtain b from $\rho_b^* = (\rho_b \otimes \widetilde{\rho_b})^{\otimes\lambda}$. For that, the adversary needs to obtain b from ρ_b or $\widetilde{\rho_b}$. Because (Q_0, Q_1) satisfies computational hiding, the QPT adversary cannot obtain b from ρ_b. Furthermore, $(\widetilde{Q_0}, \widetilde{Q_1})$ also satisfies computational hiding because $(\widetilde{Q_0}, \widetilde{Q_1})$ is a flavor conversion of (Q_0, Q_1). Therefore, the QPT adversary cannot obtain b from $\widetilde{\rho_b}$.

2.4 Universal Construction and Universal Plaintext Expansion for Unclonable Encryption

Once we have obtained a robust combiner for each primitive, we can construct a universal construction for the primitive in a relatively straightforward manner. A naive idea is to think of all descriptions of algorithms as candidates for

the primitive and combine them via a robust combiner. The combined protocol satisfies correctness and security as long as there exists the primitive since one of the candidates satisfies correctness and security when the primitive exists. Although this naive idea does not work as it is because we do not care about efficiency and each candidate may not halt, we can resolve it by modifying the idea. For details, please see the full version.

We give another universal construction for one-time unclonable SKE assuming decomposable quantum randomized encoding whose construction is inspired by [WW23]. Although we additionally use a decomposable quantum randomized encoding for this construction, we can expand the plaintext of one-time unclonable SKE. Note that it was an open problem to expand the plaintext of unclonable encryption since a standard transformation via bit-wise encryption does not work as pointed out in [AKL+22].

First, let us recall the decomposable quantum randomized encoding $\Sigma_{\mathsf{RE}} = \mathsf{RE}.(\mathsf{Enc}, \mathsf{Dec})$ given in [BY22]. In their decomposable quantum randomized encoding, $\mathsf{RE.Enc}$ takes as input a quantum circuit F, λ-length possibly quantum input q and λ-length classical input x, and outputs $\widehat{F(q,x)}$. Let $q[i]$ and $x[i]$ be the i-th qubit and bit of q and x, respectively. Decomposability guarantees that $\widehat{F(q,x)}$ can be separated into the offline encoding part $\widehat{F}_{\mathsf{off}}$ and online encoding parts $\left(\{\mathsf{lab}_i(q[i])\}_{i\in\{1,\cdots,\lambda\}}, \{\mathsf{lab}_{i+\lambda}(x[i])\}_{i\in\{1,\cdots,\lambda\}}\right)$ as follows:

$$\widehat{F(q,x)} \coloneqq \left(\widehat{F}_{\mathsf{off}}, \mathsf{lab}_1(q[1]), \cdots, \mathsf{lab}_\lambda(q[\lambda]), \mathsf{lab}_{\lambda+1}(x[1]), \cdots, \mathsf{lab}_{2\lambda}(x[\lambda])\right),$$

where $\widehat{F}_{\mathsf{off}}$ does not depend on q and x, $\mathsf{lab}_i(q[i])$ depends on only $q[i]$ for $i \in [\lambda]$ and $\mathsf{lab}_{i+\lambda}(x[i])$ depends on only $x[i]$ for $i \in [\lambda]$. $\mathsf{RE.Dec}$ takes as input $\widehat{F(q,x)}$ and outputs $F(q,x)$. The security roughly guarantees that for any quantum circuits F_1, F_2 with the same size, and any quantum and classical inputs $(\{q_1, x_1\}, \{q_2, x_2\})$ such that $F_1(q_1, x_1) = F_2(q_2, x_2)$, $\widehat{F_1(q_1, x_1)}$ is computationally indistinguishable from $\widehat{F_2(q_2, x_2)}$.

Now, we describe our one-time unclonable SKE $\Sigma_{\mathsf{Univ}} = (\mathsf{KeyGen}_{\mathsf{Univ}}, \mathsf{Enc}_{\mathsf{Univ}}, \mathsf{Dec}_{\mathsf{Univ}})$:

$\mathsf{KeyGen}_{\mathsf{Univ}}(1^\lambda)$: Our key generation algorithm $\mathsf{KeyGen}_{\mathsf{Univ}}(1^\lambda)$ first samples $x \leftarrow \{0,1\}^\lambda$. Then, it samples $R[i] \leftarrow \{0,1\}^{\ell(\lambda)}$ for $i \in [\lambda]$, and outputs $\mathsf{sk} \coloneqq (x, \{R[i]\}_{i\in[\lambda]})$. Here, $\ell(\lambda)$ is the size of online encoding of $\mathsf{RE.Enc}$.

$\mathsf{Enc}_{\mathsf{Univ}}(\mathsf{sk}, m)$: Our encryption algorithm $\mathsf{Enc}_{\mathsf{Univ}}(\mathsf{sk}, m)$ first generates a quantum circuit $C[m]$ that outputs m for any inputs, where the quantum circuit is padded to an appropriate size, which we will specify later. Then, $\mathsf{Enc}_{\mathsf{Univ}}(\mathsf{sk}, m)$ computes $\widehat{C[m]}_{\mathsf{off}}$, which is the offline encoding of $C[m]$. Next, it computes $\mathsf{lab}_i(0)$ for $i \in [\lambda]$ and $\mathsf{lab}_{\lambda+i}(b)$ for $i \in [\lambda]$ and $b \in \{0,1\}$. Finally, it samples $S[i] \leftarrow \{0,1\}^\lambda$, and computes $\mathsf{Lab.CT}[i, x[i]] = R[i] + \mathsf{lab}_{\lambda+i}(x[i])$ and $\mathsf{Lab.CT}[i, 1-x[i]] = S[i] + \mathsf{lab}_{\lambda+i}(1-x[i])$ for all $i \in [\lambda]$. The ciphertext of $\mathsf{Enc}_{\mathsf{Univ}}(\mathsf{sk}, m)$ is

$$\widehat{C[m]}_{\mathsf{off}}, \{\mathsf{lab}_i(0)\}_{i\in[\lambda]}, \{\mathsf{Lab.CT}[i,b]\}_{i\in[\lambda],b\in\{0,1\}}.$$

$\mathsf{Dec}_{\mathsf{Univ}}(\mathsf{sk}, \mathsf{CT})$: Our decryption algorithm $\mathsf{Dec}_{\mathsf{Univ}}(\mathsf{sk}, \mathsf{CT})$ works as follows. First, let $\mathsf{sk} = (x, \{R[i]\}_{i\in[\lambda]})$ and $\mathsf{CT} = \left(\widehat{C[m]}_{\mathsf{off}}, \{\mathsf{lab}_i(0)\}_{i\in[\lambda]}, \{\mathsf{Lab}.\mathsf{CT}[i,b]\}_{i\in[\lambda],b\in\{0,1\}}\right)$. $\mathsf{Dec}_{\mathsf{Univ}}(\mathsf{sk}, \mathsf{CT})$ first computes $\mathsf{lab}_{\lambda+i}(x[i]) = R[i] + \mathsf{Lab}.\mathsf{CT}[i, x[i]]$ for all $i \in [\lambda]$, and runs $\mathsf{RE}.\mathsf{Dec}(\widehat{C[m]}_{\mathsf{off}}, \{\mathsf{lab}_i(0)\}_{i\in[\lambda]}, \{\mathsf{lab}_{i+\lambda}(x[i])\}_{i\in[\lambda]})$.

Clearly, our encryption algorithm can encrypt arbitrary-length plaintext. We can see that our construction satisfies correctness. More formally, $\mathsf{Dec}_{\mathsf{Univ}}(\mathsf{sk}, \mathsf{CT}_m)$ outputs m with high probability if $\mathsf{sk} \leftarrow \mathsf{KeyGen}_{\mathsf{Univ}}(1^\lambda)$ and $\mathsf{CT}_m \leftarrow \mathsf{Enc}_{\mathsf{Univ}}(\mathsf{sk}, m)$. From our construction, $\mathsf{Dec}_{\mathsf{Univ}}(\mathsf{sk}, \mathsf{CT}_m)$ outputs the output of $\mathsf{RE}.\mathsf{Dec}(\widehat{C[m]}_{\mathsf{off}}, \{\mathsf{lab}_i(0)\}_{i\in[\lambda]}, \{\mathsf{lab}_{i+\lambda}(x[i])\}_{i\in[\lambda]})$, where $\left(\widehat{C[m]}_{\mathsf{off}}, \{\mathsf{lab}_i(0)\}_{i\in[\lambda]}, \{\mathsf{lab}_{i+\lambda}(x[i])\}_{i\in[\lambda]}\right) \leftarrow \mathsf{RE}.\mathsf{Enc}(C, 0^\lambda, x)$. From the correctness of decomposable quantum randomized encoding, $\mathsf{RE}.\mathsf{Dec}(\widehat{C[m]}_{\mathsf{off}}, \{\mathsf{lab}_i(0)\}_{i\in[\lambda]}, \{\mathsf{lab}_{i+\lambda}(x[i])\}_{i\in[\lambda]})$ outputs $C[m](0^\lambda, x)$, which is equal to m.

Furthermore, our construction Σ_{Univ} satisfies unclonable IND-CPA security as long as the underlying decomposable quantum randomized encoding Σ_{RE} satisfies security and there exists a one-time unclonable SKE for single-bit plaintexts. To see this, we introduce some notations and observations. We write $\Sigma_{\mathsf{unclone}} = \mathsf{Unclone}.(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ to mean a one-time unclonable SKE for single-bit plaintexts, which we assume to exist. Without loss of generality, we can assume that the secret key sk generated by $\mathsf{Unclone}.\mathsf{KeyGen}(1^\lambda)$ is uniformly randomly sampled and $|\mathsf{sk}| = |\mathsf{CT}| = \lambda$ for all security parameters λ. Moreover, we can assume that for a security parameter λ, $\mathsf{Unclone}.\mathsf{Dec}(\mathsf{sk}, \mathsf{CT})$ is a quantum algorithm that runs some quantum circuit $\mathsf{Unclone}.\mathsf{Dec}_\lambda$ on CT and sk, and outputs its output. We introduce a quantum circuit $D_\lambda[m_0, m_1]$ that takes as input CT and sk, and runs the quantum circuit $\mathsf{Unclone}.\mathsf{Dec}_\lambda$ on CT and sk, obtains b and outputs m_b. The size of $C[m]$ is padded so that its size is equal to $D_\lambda[m_0, m_1]$.

Now, we can see that our construction Σ_{Univ} satisfies one-time unclonable IND-CPA security. In the first step of the proof, we switch the following real ciphertext for message m_b

$$\mathsf{CT}_b = \left(\widehat{C[m_b]}_{\mathsf{off}}, \{\mathsf{lab}_i(0)\}_{i\in[\lambda]}, \{\mathsf{Lab}.\mathsf{CT}[i,\beta]\}_{i\in[\lambda],\beta\in\{0,1\}}\right)$$

to the following modified ciphertext

$$\widetilde{\mathsf{CT}_b} = \left(\widehat{D[m_0, m_1]}_{\mathsf{off}}, \{\mathsf{lab}_i(\mathsf{unclone}.\mathsf{CT}_b[i])\}_{i\in[\lambda]}, \{\mathsf{Lab}.\mathsf{CT}[i,\beta]\}_{i\in[\lambda],\beta\in\{0,1\}}\right),$$

where $\mathsf{unclone}.\mathsf{CT}_b \leftarrow \mathsf{Unclone}.\mathsf{Enc}(x, b)$ and $\mathsf{unclone}.\mathsf{CT}_b[i]$ is the i-th qubit of $\mathsf{unclone}.\mathsf{CT}_b$ and $x \leftarrow \{0,1\}^\lambda$. This change does not affect the output of the security experiment because Σ_{RE} satisfies security and we have

$$D[m_0, m_1](\mathsf{unclone}.\mathsf{CT}_b, x) = C[m_b](0^\lambda, x) = m_b.$$

In the next step, we can reduce the security of our construction Σ_{Univ} to that of one-time unclonable SKE for single-bit plaintexts $\Sigma_{\mathsf{unclone}}$. This is because the adversary $(\mathcal{A}, \mathcal{B}, \mathcal{C})$ of $\Sigma_{\mathsf{unclone}}$ can simulate the challenger of Σ_{Univ} sicne $(\mathcal{A}, \mathcal{B}, \mathcal{C})$ can simulate $\widetilde{\mathsf{CT}_b}$ by using $\mathsf{unclone}.\mathsf{CT}_b$.

3 Preliminaries

3.1 Notations

Here we introduce basic notations we will use in this paper. $x \leftarrow X$ denotes selecting an element x from a finite set X uniformly at random, and $y \leftarrow \mathcal{A}(x)$ denotes assigning to y the output of a quantum or probabilistic or deterministic algorithm $\mathcal{A}$ on an input x. When we explicitly write that $\mathcal{A}$ uses randomness r, we write $y \leftarrow \mathcal{A}(x;r)$. Let $[n] := \{1, \cdots, n\}$. For $x \in \{0,1\}^n$ and $i \in [n]$, x_i and $x[i]$ are the i-th bit value of x. For an n-qubit state ρ and $i \in [n]$, we write ρ_i and $\rho[i]$ to mean a quantum state that traces out all states other than the i-th qubit of ρ. QPT stands for quantum polynomial time. A function $f : \mathbb{N} \to \mathbb{R}$ is a negligible function if, for any constant c, there exists $\lambda_0 \in \mathbb{N}$ such that for any $\lambda > \lambda_0$, $f(\lambda) < 1/\lambda^c$. We write $f(\lambda) \leq \mathsf{negl}(\lambda)$ to denote $f(\lambda)$ being a negligible function.

For simplicity, we often write $|0\rangle$ to mean $|0 \cdots 0\rangle$. For any two quantum states ρ_1 and ρ_2, $F(\rho_1, \rho_2)$ is the fidelity between them, and $\mathsf{TD}(\rho_1, \rho_2)$ is the trace distance between them.

For a quantum algorithm $\mathcal{A}$, and quantum states ρ and σ, we say that $\mathcal{A}$ distinguishes ρ from σ with advantage Δ if

$$|\Pr[1 \leftarrow \mathcal{A}(\rho)] - \Pr[1 \leftarrow \mathcal{A}(\sigma)]| = \Delta.$$

We say that ρ is c-computationally indistinguishable (resp. c-statistically indistinguishable) from σ if no QPT algorithms (resp. unbounded algorithms) can distinguish ρ from σ with advantage greater than c.

3.2 Cryptographic Tools

In this section, we introduce cryptographic tools which we will use.

Canonical Quantum Bit Commitment.

Definition 3.1 (Canonical Quantum Bit Commitment [Yan22]). *A candidate for canonical quantum bit commitment is a set of uniform QPT unitaries $\{Q_0(\lambda), Q_1(\lambda)\}_{\lambda\in\mathbb{N}}$ acting on the register $\mathbf{C}$ and $\mathbf{R}$. We consider the following two properties.*

Hiding. We say that a candidate for canonical quantum bit commitment $\{Q_0(\lambda), Q_1(\lambda)\}_{\lambda\in\mathbb{N}}$ satisfies c-statistical hiding (resp. c-computational hiding) if $\mathrm{Tr}_{\mathbf{R}}(Q_0(\lambda)|0\rangle_{\mathbf{CR}})$ is c-statistically indistinguishable (resp. c-computationally indistinguishable) from $\mathrm{Tr}_{\mathbf{R}}(Q_1(\lambda)|0\rangle_{\mathbf{CR}})$ for all sufficiently large $\lambda \in \mathbb{N}$.

If a candidate for canonical quantum bit commitment satisfies $\mathsf{negl}(\lambda)$-statistical hiding (resp. $\mathsf{negl}(\lambda)$-computational hiding), then we say that the candidate satisfies statistical hiding (resp. computational hiding).

Binding. We say that a candidate for canonical quantum bit commitment $\{Q_0(\lambda), Q_1(\lambda)\}_{\lambda\in\mathbb{N}}$ satisfies c-statistical binding (resp. c-computational binding)

if for all sufficiently large security parameters $\lambda \in \mathbb{N}$, any unbounded-time (resp. QPT) unitary U over $\mathbf{R}$ and an additional register $\mathbf{Z}$ and any polynomial-size $|\tau\rangle$, it holds that

$$\left\| (\langle 0|Q_1^\dagger(\lambda))_{\mathbf{C},\mathbf{R}} (I_{\mathbf{C}} \otimes U_{\mathbf{R},\mathbf{Z}}) ((Q_0(\lambda)|0\rangle_{\mathbf{C},\mathbf{R}})|\tau\rangle_{\mathbf{Z}}) \right\| \leq c.$$

If a candidate for canonical quantum bit commitment satisfies $\mathsf{negl}(\lambda)$-statistical binding (resp. $\mathsf{negl}(\lambda)$-computational binding), then we say that the candidate satisfies statistical binding (resp. computational binding).

It was shown that we can convert the flavor of quantum bit commitment as follows.

Lemma 3.1 (Converting Flavors:[HMY23]). *Let $\{Q_0(\lambda), Q_1(\lambda)\}_{\lambda\in\mathbb{N}}$ be a candidate of canonical quantum bit commitment. Let $\{\widetilde{Q_0}(\lambda), \widetilde{Q_1}(\lambda)\}_{\lambda\in\mathbb{N}}$ be a candidate of canonical quantum bit commitment described as follows:*

- *The role of commitment and reveal registers are swapped from $(Q_0(\lambda), Q_1(\lambda))$ and the commitment register is augmented by an additional one-qubit register which we denote $\mathbf{D}$. In other words, if $\mathbf{C}$ and $\mathbf{R}$ are the commitment and reveal registers of $(Q_0(\lambda), Q_1(\lambda))$, then the commitment and reveal registers of $(\widetilde{Q_0}(\lambda), \widetilde{Q_1}(\lambda))$ are defined as $\widetilde{\mathbf{C}} := (\mathbf{R}, \mathbf{D})$ and $\widetilde{\mathbf{R}} := \mathbf{C}$, where $\mathbf{D}$ is an additional one-qubit register.*
- *For $b \in \{0,1\}$, the unitary $\widetilde{Q_b}(\lambda)$ is defined as follows:*

$$\widetilde{Q_b}(\lambda) := (Q_0(\lambda) \otimes |0\rangle\langle 0|_{\mathbf{D}} + Q_1(\lambda) \otimes |1\rangle\langle 1|_{\mathbf{D}}) \left(I_{\mathbf{RC}} \otimes Z_{\mathbf{D}}^b H_{\mathbf{D}}\right).$$

The following holds for $X, Y \in \{$statistical, computational$\}$.

1. *If $\{Q_0(\lambda), Q_1(\lambda)\}_{\lambda\in\mathbb{N}}$ satisfies c-X hiding, then $\{\widetilde{Q_0}(\lambda), \widetilde{Q_1}(\lambda)\}_{\lambda\in\mathbb{N}}$ satisfies $\sqrt{c}$-X binding.*
2. *If $\{Q_0(\lambda), Q_1(\lambda)\}_{\lambda\in\mathbb{N}}$ satisfies $\mathsf{negl}(\lambda)$-Y binding, then $\{\widetilde{Q_0}(\lambda), \widetilde{Q_1}(\lambda)\}_{\lambda\in\mathbb{N}}$ satisfies $\mathsf{negl}(\lambda)$-Y hiding.*

Unclonable Encryption. In this work, we consider unclonable encryption with unclonable IND-CPA security.

Definition 3.2 (Unclonable Secret-Key Encryption [BL20]). *A candidate for unclonable secret-key encryption for $n(\lambda)$-bit plaintexts is a set of algorithms $\Sigma := (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ such that:*

$\mathsf{KeyGen}(1^\lambda)$*: It takes as input a security parameter 1^λ, and outputs a classical secret-key sk.*

$\mathsf{Enc}(1^\lambda, \mathsf{sk}, m)$*: It takes as input a security parameter 1^λ, sk and $m \in \{0,1\}^{n(\lambda)}$, and outputs a quantum ciphertext CT.*

$\mathsf{Dec}(1^\lambda, \mathsf{sk}, \mathsf{CT})$*: It takes as input a security parameter 1^λ, sk and CT, and outputs m.*

We say that a candidate Σ is an unclonable SKE scheme if it satisfies the following efficiency, correctness, IND-CPA security, and unclonable IND-CPA security.

Efficiency. The algorithms $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *are uniform QPT algorithms.*

Correctness. We have

$$\Pr\left[m \leftarrow \mathsf{Dec}(1^\lambda, \mathsf{sk}, \mathsf{CT}) : \mathsf{sk} \leftarrow \mathsf{KeyGen}(1^\lambda), \mathsf{CT} \leftarrow \mathsf{Enc}(1^\lambda, \mathsf{sk}, m)\right] \geq 1 - \mathsf{negl}(\lambda).$$

Unclonable IND-CPA Security. We require that Σ *satisfies standard IND-CPA security. In addition to the standard IND-CPA security, we require that* Σ *satisfies the unclonable IND-CPA security defined below. Given an unclonable encryption* Σ*, we consider the unclonable IND-CPA security experiment* $\mathsf{Exp}^{\mathsf{unclone}}_{\Sigma,(\mathcal{A},\mathcal{B},\mathcal{C})}(\lambda)$ *against* $(\mathcal{A}, \mathcal{B}, \mathcal{C})$*.*

1. *The challenger runs* $\mathsf{sk} \leftarrow \mathsf{KeyGen}(1^\lambda)$*.*
2. $\mathcal{A}$ *can query* $\mathsf{Enc}(1^\lambda, \mathsf{sk}, \cdot)$ *polynomially many times.*
3. $\mathcal{A}$ *sends* (m_0, m_1) *to the challenger.*
4. *The challenger samples* $b \leftarrow \{0,1\}$*, runs* $\mathsf{CT}_b \leftarrow \mathsf{Enc}(1^\lambda, \mathsf{sk}, m_b)$*, and sends* CT_b *to* $\mathcal{A}$*.*
5. $\mathcal{A}$ *produces* $\rho_{\mathcal{B},\mathcal{C}}$ *and sends the corresponding registers to* $\mathcal{B}$ *and* $\mathcal{C}$*.*
6. $\mathcal{B}$ *and* $\mathcal{C}$ *receive* sk *and output* $b_\mathcal{B}$ *and* $b_\mathcal{C}$*.*
7. *The experiment outputs* 1 *indicating win if* $b_\mathcal{B} = b_\mathcal{C} = b$*, and otherwise* 0*.*

We say that Σ *is unclonable IND-CPA secure if for all sufficiently large security parameters* $\lambda \in \mathbb{N}$*, for all non-uniform QPT adversaries* $(\mathcal{A}, \mathcal{B}, \mathcal{C})$*,*

$$\Pr\left[\mathsf{Exp}^{\mathsf{unclone}}_{\Sigma,(\mathcal{A},\mathcal{B},\mathcal{C})}(\lambda) = 1\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda).$$

Decomposable Quantum Randomized Encoding.

Definition 3.3 (Decomposable Quantum Randomized Encoding (DQRE) [BY22]). *A DQRE scheme is a tuple of algorithms* $(\mathsf{Enc}, \mathsf{Dec})$ *such that:*

$\mathsf{Enc}(1^\lambda, F, x)$*: It takes* 1^λ *with* $\lambda \in \mathbb{N}$*, a general quantum circuit* F *and a possibly quantum input* x *as inputs, and outputs* $\widehat{F(x)}$*.*

$\mathsf{Dec}(1^\lambda, \widehat{F(x)})$*: It takes as input* 1^λ*, and* $\widehat{F(x)}$*, and outputs* $F(x)$*.*

We require the following four properties:

Efficiency. $(\mathsf{Enc}, \mathsf{Dec})$ *are uniform QPT algorithms.*

Correctness. For all quantum states (x, q) *and randomness* r*, it holds that* $(F(x), q) = (\mathsf{Dec}(1^\lambda, \widehat{F}(x; r)), q)$*, where* $\widehat{F}(x; r)$ *is an output of* $\mathsf{Enc}(1^\lambda, F, x; r)$*.*

Security. There exists a uniform QPT algorithm Sim *such that for all quantum states* (x, q) *and non-uniform QPT adversary* $\mathcal{A}$*, there exists some negligible function* negl *that satisfies,*

$$|\Pr\left[1 \leftarrow \mathcal{A}(\widehat{F}(x;r), q)\right] - \Pr\left[1 \leftarrow \mathcal{A}(\mathsf{Sim}(1^\lambda, |F|, F(x)), q)\right]| \leq \mathsf{negl}(\lambda),$$

where the state on the left-hand side is averaged over r *and* $|F|$ *is the size of the general quantum circuit* F*.*

Remark 3.1. In the security of the original paper [BY22], the simulator Sim takes the topology of F as input. Without loss of generality, we can replace the topology of F with the size of F because we can hide the topology of F by using a universal quantum circuit.

Decomposability. There exists a quantum state e (called the resource state of the encoding), and operation $\widehat{F}_{\mathsf{off}}$ (called the offline part of the encoding) and a collection of input encoding operations $\widehat{F_1}, \cdots, \widehat{F_n}$ such that for all inputs $x = (x_1, \cdots, x_n)$,

$$\widehat{F}(x;r) = \left(\widehat{F}_{\mathsf{off}}, \widehat{F}_1, \widehat{F_2}, \cdots, \widehat{F_n}\right)(x, r, e)$$

where the functions $\widehat{F}_{\mathsf{off}}, \widehat{F_1}, \cdots, \widehat{F_n}$ act on disjoint subsets of qubits from e, x (but can depend on all bits of r), each $\widehat{F}_i$ acts on a single qubit x_i and $\widehat{F}_{\mathsf{off}}$ does not act on any of the qubits of x.

Classical Labels. If x_i is a classical bit, then $\widehat{F}_i(x_i, r)$ is a classical string as well.

Theorem 3.1 ([BY22])**.** *Decomposable quantum randomized encoding exists if OWFs exist.*

Proposition 3.1. *Let* $\Sigma := (\mathsf{Enc}, \mathsf{Dec})$ *be a decomposable quantum randomized encoding. Then, for any quantum circuits* F_0, F_1 *with the same size, for any possibly quantum input* x_0 *and* x_1 *such that* $F_0(x_0) = F_1(x_1)$*,* $\widehat{F_0}(x_0; r_0)$ *is computationally indistinguishable from* $\widehat{F_1}(x_1; r_1)$*, where both quantum states are averaged over the randomness* r_0 *and* r_1*.*

This can be shown by a standard hybrid argument, and thus we omit the proof.

4 Robust OWSGs Combiner

Based on the idea Sect. 2.1, we construct robust OWSGs combiner and construct universal construction for OWSGs. For details, see the full version.

5 Robust Combiner for Public-Key Quantum Money Mini-Scheme

Based on the idea Sect. 2.1, we construct robust combiner for public key quantum money mini-scheme and construct universal construction for it. For details, see the full version.

6 Robust Canonical Quantum Bit Commitment Combiner

Definition 6.1 (Robust Canonical Quantum Bit Commitment Combiner). *A robust canonical quantum bit commitment combiner is a deterministic classical polynomial-time Turing machine $\mathcal{M}$ with the following properties:*

- *$\mathcal{M}$ takes as input 1^n and n-deterministic classical polynomial-time Turing machine $\{\mathcal{T}_i\}_{i\in[n]}$ that produces unitary, and outputs a deterministic classical polynomial-time Turing machine $\mathcal{T}$ that produces unitary.*
- *Let $(Q_{i,0}(\lambda), Q_{i,1}(\lambda))$ be the unitary obtained by $\mathcal{T}_i(\lambda)$ and let $(Q_0(\lambda), Q_1(\lambda))$ be the unitary obtained by $\mathcal{T}(\lambda)$. If one of $\{\{Q_{i,0}(\lambda), Q_{i,1}(\lambda)\}_{\lambda\in\mathbb{N}}\}_{i\in[n]}$ satisfies computational binding and computational hiding, then $\{Q_0(\lambda), Q_1(\lambda)\}_{\lambda\in\mathbb{N}}$ is a quantum bit commitment that satisfies statistical binding and computational hiding.*

In this section, we show the Theorem 6.1.

Theorem 6.1. *There exists a robust canonical quantum bit commitment combiner.*

First, let us introduce the following Proposition 6.1.

Proposition 6.1. *Let $\Sigma = \{Q_0(\lambda), Q_1(\lambda)\}_{\lambda\in\mathbb{N}}$ be a candidate of a canonical quantum bit commitment. From Σ, we can construct a canonical quantum bit commitment $\Sigma^* := \{Q_0^*(\lambda), Q_1^*(\lambda)\}_{\lambda\in\mathbb{N}}$ such that:*

1. *Σ^* satisfies statistical binding.*
2. *If Σ satisfies computational binding and computational hiding, then Σ^* satisfies computational hiding.*

Proposition 6.1 directly follows from the following Lemma 6.1.

Lemma 6.1 (Amplifying Binding). *Let $\{Q_0(\lambda), Q_1(\lambda)\}_{\lambda\in\mathbb{N}}$ be a candidate of canonical quantum bit commitment. Let $\{Q_0^*(\lambda), Q_1^*(\lambda)\}_{\lambda\in\mathbb{N}}$ be a candidate of canonical quantum bit commitment described as follows:*

- *If $\mathbf{C}$ and $\mathbf{R}$ are the commitment and reveal registers of $(Q_0(\lambda), Q_1(\lambda))$, and $\widetilde{\mathbf{C}}$ and $\widetilde{\mathbf{R}}$ are the commitment and reveal registers of $(\widetilde{Q_0}(\lambda), \widetilde{Q_1}(\lambda))$, which is the flavor conversion of $(Q_0(\lambda), Q_1(\lambda))$ introduced in Lemma 3.1, then the commitment and reveal registers of $(Q_0^*(\lambda), Q_1^*(\lambda))$ are defined as $\mathbf{C}^* := \left(\mathbf{C}^{\otimes\lambda}, \widetilde{\mathbf{C}}^{\otimes\lambda}\right)$, and $\mathbf{R}^* := \left(\mathbf{R}^{\otimes\lambda}, \widetilde{\mathbf{R}}^{\otimes\lambda}\right)$.*

– *For $b \in \{0,1\}$, the unitary $Q_b^*(\lambda)$ is defined as follows:*

$$Q_b^*(\lambda) \coloneqq (Q_b(\lambda) \otimes \widetilde{Q_b}(\lambda))^{\otimes \lambda}.$$

Then, the following is satisfied:

1. *$\{Q_0^*(\lambda), Q_1^*(\lambda)\}_{\lambda\in\mathbb{N}}$ satisfies statistical binding.*
2. *If $\{Q_0(\lambda), Q_1(\lambda)\}_{\lambda\in\mathbb{N}}$ satisfies computational hiding and computational binding, then $\{Q_0^*(\lambda), Q_1^*(\lambda)\}_{\lambda\in\mathbb{N}}$ satisfies computational hiding.*

Proof of Lemma 6.1. Below, we fix the security parameter λ, and write (Q_0, Q_1), $(\widetilde{Q_0}, \widetilde{Q_1})$ and (Q_0^*, Q_1^*) to mean $(Q_0(\lambda), Q_1(\lambda))$, $(\widetilde{Q_0}(\lambda), \widetilde{Q_1}(\lambda))$ and $(Q_0^*(\lambda), Q_1^*(\lambda))$, respectively.

Proof of the First Item. We define

$$\rho_b \coloneqq \mathrm{Tr}_{\mathbf{R}}(Q_b|0\rangle_{\mathbf{C},\mathbf{R}}) \text{ and } \widetilde{\rho_b} \coloneqq \mathrm{Tr}_{\widetilde{\mathbf{R}}}(\widetilde{Q_b}|0\rangle_{\widetilde{\mathbf{C}},\widetilde{\mathbf{R}}}) \text{ and } \rho_b^* \coloneqq \mathrm{Tr}_{\mathbf{R}^*}(Q_b^*|0\rangle_{\mathbf{C}^*,\mathbf{R}^*}).$$

From the construction of Q_0^* and Q_1^*, we have

$$(\rho_b \otimes \widetilde{\rho_b})^{\otimes\lambda} \coloneqq \mathrm{Tr}_{\mathbf{R}^*}(Q_b^*|0\rangle_{\mathbf{C}^*,\mathbf{R}^*}).$$

Let $0 \le f \le 1$ be some value such that

$$F(\rho_0, \rho_1) = f.$$

We have

$$\mathsf{TD}(\rho_0, \rho_1) \le \sqrt{1 - F(\rho_0, \rho_1)} \le \sqrt{1-f}.$$

In particular, this implies that (Q_0, Q_1) satisfies $\sqrt{1-f}$-statistical hiding. From Lemma 3.1, this implies that $(\widetilde{Q_0}, \widetilde{Q_1})$ satisfies $(1-f)^{1/4}$-statistical binding. Furthermore, from Uhlmann's theorem, we have

$$F(\widetilde{\rho_0}, \widetilde{\rho_1}) \le (1-f)^{1/2}.$$

Therefore, we have

$$F(\rho_0^*, \rho_1^*) = F\left((\rho_0 \otimes \widetilde{\rho_0})^{\otimes\lambda}, (\rho_1 \otimes \widetilde{\rho_1})^{\otimes\lambda}\right) \le F(\rho_0, \rho_1)^\lambda F(\widetilde{\rho_0}, \widetilde{\rho_1})^\lambda \le f^\lambda (1-f)^{\lambda/2} \le 2^{-\lambda/2},$$

which implies that (Q_0^*, Q_1^*) satisfies statistical binding.

Proof of the Second Item. We prove that (Q_0^*, Q_1^*) satisfies computational hiding if (Q_0, Q_1) satisfies computational hiding and computational binding. Because $(\widetilde{Q_0}, \widetilde{Q_1})$ is the flavor conversion of (Q_0, Q_1), $(\widetilde{Q_0}, \widetilde{Q_1})$ also satisfies computational hiding. Therefore, we can reduce the computational hiding of (Q_0^*, Q_1^*) to those of (Q_0, Q_1) and $(\widetilde{Q_0}, \widetilde{Q_1})$ by a standard hybrid argument. □

Proof of Theorem 6.1. Below, we consider some fixed constant n. For $i \in [n]$, let $\mathcal{T}_i$ be a deterministic classical Turing machine that takes as input 1^λ, and outputs $(Q_{i,0}(\lambda), Q_{i,1}(\lambda))$. Let $\Sigma_i \coloneqq \{Q_{i,0}(\lambda), Q_{i,1}(\lambda)\}_{\lambda\in\mathbb{N}}$ be a candidate of canonical quantum bit commitment. Let $\Sigma_i^* \coloneqq \{Q_{i,0}^*(\lambda), Q_{i,1}^*(\lambda)\}_{\lambda\in\mathbb{N}}$ be a candidate of canonical quantum bit commitment such that:

1. Σ_i^* satisfies statistical binding.
2. Σ_i^* satisfies computational hiding if Σ_i satisfies computational hiding and computational binding.

Note that such a canonical quantum bit commitment is obtained from Proposition 6.1.

A robust canonical quantum bit commitment combiner is a deterministic classical polynomial-time Turing machine $\mathcal{M}$ that takes as input 1^n and $\{\mathcal{T}_i\}_{i\in[n]}$, and outputs a deterministic classical polynomial-time Turing machine $\mathcal{T}$ that works as follows. $\mathcal{T}$ takes as input 1^λ and outputs the following QPT unitary $(\mathsf{Comb}.Q_0(\lambda), \mathsf{Comb}.Q_1(\lambda))$:

- If $\mathbf{C}_i^*$ and $\mathbf{R}_i^*$ are the commitment register and the reveal register of $(Q_{i,0}^*(\lambda), Q_{i,1}^*(\lambda))$, then the commitment and reveal register of $(\mathsf{Comb}.Q_0(\lambda), \mathsf{Comb}.Q_1(\lambda))$ are defined as $\mathbf{C} := \{\mathbf{C}_i^*\}_{i\in[n]}$ and $\mathbf{R} = (\{\mathbf{R}_i^*\}_{i\in[n]}, \{\mathbf{D}_i^*\}_{i\in[n]})$, where $\mathbf{D}_i^*$ is an additional one-qubit register for $i \in [n]$.
- For $b \in \{0,1\}$, the unitary $\mathsf{Comb}.Q_b$ is defined as follows:

$$\mathsf{Comb}.Q_b(\lambda) := \left(\sum_{r\in\{0,1\}^n} \bigotimes_{i\in[n]} (Q_{i,r_i}^*(\lambda) \otimes |r_i\rangle\langle r_i|_{\mathbf{D}_i^*})\right) \cdot \left(\bigotimes_{i\in[n]} I_{\mathbf{C}_i^*,\mathbf{R}_i^*} \otimes X_{\mathbf{D}_1^*}^b \bigotimes_{i\in\{2,\cdots,n\}} \mathsf{CNOT}_{\mathbf{D}_1^*,\mathbf{D}_i^*} \bigotimes_{i\in\{2,\cdots,n\}} H_{\mathbf{D}_i^*}\right).$$

Here, r_i is the i-th bit of r and $\mathsf{CNOT}_{\mathbf{D}_1^*,\mathbf{D}_i^*}$ is a CNOT gate, where $\mathbf{D}_1^*$ is a target register and $\mathbf{D}_i^*$ is a control register. Note that we have

$$\mathsf{Comb}.Q_b(\lambda)|0\rangle_{\mathbf{C},\mathbf{R}} = \frac{1}{2^{(n-1)/2}} \sum_{\{r:\sum_{i\in[n]} r_i = b\}} \bigotimes_{i\in[n]} (Q_{i,r_i}^*(\lambda)|0\rangle_{\mathbf{C}_i^*,\mathbf{R}_i^*} \otimes |r_i\rangle_{\mathbf{D}_i^*}).$$

We have the following Lemmata 6.2 and 6.3. Therefore, Theorem 6.1 holds.

Lemma 6.2. $\{\mathsf{Comb}.Q_0(\lambda), \mathsf{Comb}.Q_1(\lambda)\}_{\lambda\in\mathbb{N}}$ *satisfies statistical binding.*

Intuitively, $\mathsf{Comb}, Q_b(\lambda)$ randomly samples $\{r[i]\}_{i\in[n]}$ such that $\sum_{i\in[n]} r[i] = b$, and commits each $r[i]$ using $Q_{i,r[i]}^*$. In order to break the statistical binding of $\{\mathsf{Comb}.Q_0(\lambda), \mathsf{Comb}.Q_1(\lambda)\}_{\lambda\in\mathbb{N}}$, the adversary needs to change one of $\{r[i]\}_{i\in[n]}$ after sending the commitment register. However, the adversary cannot do this because all of $\{Q_{i,0}^*, Q_{i,1}^*\}_{i\in[n]}$ satisfy statistical binding. For details, please see the full version.

Lemma 6.3. *If one of* $\{\Sigma_i\}_{i\in[n]}$ *satisfies computational hiding and computational binding, then* $\{\mathsf{Comb}.Q_0(\lambda), \mathsf{Comb}.Q_1(\lambda)\}_{\lambda\in\mathbb{N}}$ *satisfies computational hiding.*

The adversary of $(\mathsf{Comb}.Q_0, \mathsf{Comb}.Q_1)$ needs to obtain all of $\{r[i]\}_{i\in[n]}$ from the commitment register of $(Q_0[1], Q_1[1])$ and $\{Q^*_{i,0}, Q^*_{i,1}\}_{i\in[n]}$, but the adversary cannot do this because one of $\{Q^*_{i,0}, Q^*_{i,1}\}_{i\in[n]}$ satisfies computational hiding. For details, please see the full version. □

6.1 Universal Construction

Definition 6.2. *We say that a sequence of uniform QPT unitaries* $\Sigma_{\mathsf{Univ}} = \{Q_0(\lambda), Q_1(\lambda)\}_{\lambda\in\mathbb{N}}$ *is a universal construction of canonical quantum bit commitment if* Σ_{Univ} *is canonical quantum bit commitment as long as there exists canonical quantum bit commitment.*

Theorem 6.2. *There exists a universal construction of canonical quantum bit commitment.*

We omit the proof. For details, please see the full version.

7 Robust Combiner for Unclonable Encryption

Definition 7.1 (Robust Combiner for Unclonable Secret-Key Encryption). *A robust combiner for (one-time) unclonable secret-key encryption with* $\ell(\lambda)$*-bit plaintexts is a deterministic classical polynomial-time Turing machine* $\mathcal{M}$ *with the following properties:*

- $\mathcal{M}$ *takes as input* 1^n *with* $n \in \mathbb{N}$ *and* n*-candidates (one-time) unclonable secret-key encryption with* $\ell(\lambda)$*-bit plaintexts* $\{\Sigma_i := (\mathsf{KeyGen}_i, \mathsf{Enc}_i, \mathsf{Dec}_i)\}_{i\in[n]}$ *promised that all candidates satisfies efficiency, and outputs a set of algorithms* $\Sigma := (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$.
- *If all of* $\{\Sigma_i\}_{i\in[n]}$ *satisfies efficiency and at least one of* $\{\Sigma_i\}_{i\in[n]}$ *satisfies correctness, (one-time) IND-CPA security and (one-time) unclonable IND-CPA security, then* Σ *is (one-time) unclonable secret-key encryption for* $\ell(\lambda)$*-bit plaintexts that satisfies efficiency, correctness, (one-time) IND-CPA security and (one-time) unclonable IND-CPA security.*

In this section, we prove the following Theorem 7.1.

Theorem 7.1. *There exists a robust combiner for (one-time) unclonable secret-key encryption with* $\ell(\lambda)$*-bit plaintexts for all polynomial* ℓ.

As a corollary, we obtain the following Corollary 7.1.

Corollary 7.1. *There exists a robust combiner for unclonable public-key encryption with* $\ell(\lambda)$*-bit plaintexts for all polynomial* ℓ.

Proof of Corollary 7.1. We give a rough sketch of the proof.

Corollary 7.1 follows from the following observations. We can trivially obtain one-time unclonable SKE from unclonable PKE. From Theorem 7.1, we have a robust combiner for one-time unclonable SKE. Furthermore, we can trivially construct PKE with quantum ciphertexts from unclonable PKE. It is known that there exists a robust PKE combiner [HKN+05], and we observe that we can also construct a robust combiner for PKE with quantum ciphertexts in the same way. Moreover, we can construct unclonable PKE from one-time unclonable SKE, and PKE with quantum ciphertexts. This is because we can construct unclonable PKE from one-time SKE and receiver non-committing encryption with quantum ciphertexts [4] (For the detail, please see the full version), and receiver non-committing encryption with quantum ciphertexts can be constructed from PKE with quantum ciphertexts in the same way as the classical ciphertext case [CHK05, KNTY19].

By combining these observations, we can construct a robust combiner for unclonable PKE as follows. Given candidates of unclonable PKE $\{\Sigma_i\}_{i\in[n]}$, we first use a robust combiner for one-time unclonable SKE, and obtain a new candidate of one-time unclonable SKE Σ_{SKE} regarding each candidate Σ_i as a one-time unclonable SKE scheme. Next, we use a robust combiner for PKE with quantum ciphertexts and obtain a new candidate of PKE with quantum ciphertexts Σ_{PKE} regarding each candidate Σ_i as a (not necessarily unclonable) PKE scheme. Then, we construct a receiver non-committing encryption with quantum ciphertexts Σ_{NCE} from Σ_{PKE}. Finally, we construct unclonable PKE $\Sigma_{\mathsf{unclone}}$ from one-time unclonable SKE Σ_{SKE} and receiver non-committing encryption with quantum ciphertexts Σ_{NCE}. □

For proving Theorem 7.1, we introduce the following Lemma 7.1.

Lemma 7.1. *Let Σ be a candidate for (one-time) unclonable secret-key encryption with $\ell(\lambda)$-bit plaintexts. From Σ, we can construct a (one-time) unclonable secret-key encryption with $\ell(\lambda)$-bit plaintexts $\Sigma^* := (\mathsf{KeyGen}^*, \mathsf{Enc}^*, \mathsf{Dec}^*)$ such that:*

1. *Σ^* is a uniform QPT algorithm, if Σ is a uniform QPT algorithm.*
2. *Σ^* satisfies perfect correctness.*
3. *Σ^* satisfies (one-time) IND-CPA security and (one-time) unclonable IND-CPA security if Σ is a uniform QPT algorithm and satisfies correctness, (one-time) IND-CPA security and (one-time) unclonable IND-CPA security.*

We omit the proof. For details, please see the full version.

[4] [AK21] shows that unclonable PKE can be constructed from one-time unclonable SKE and PKE with classical ciphertexts. Note that it is unclear whether we can construct unclonable PKE from one-time SKE and PKE with "quantum" ciphertexts in the same way as [AK21]. This is because they use the existence of OWFs in their proof although it is unclear whether PKE with quantum ciphertexts implies OWFs. Therefore, we use the technique of [HMNY21] instead. (For the detail, please see the full version).

Proof of Theorem 7.1. Below, we consider a fixed constant n and a fixed polynomial ℓ. Let us describe some notations:

Notations.

- Let Σ_i be a candidate of (one-time) unclonable secret-key encryption with $\ell(\lambda)$-length for $i \in [n]$.
- For a candidate of (one-time) unclonable secret-key encryption with $\ell(\lambda)$-bit plaintexts Σ_i, let $\Sigma_i^* := (\mathsf{KeyGen}_i^*, \mathsf{Enc}_i^*, \mathsf{Dec}_i^*)$ be a candidate of (one-time) unclonable secret-key encryption with $\ell(\lambda)$-bit plaintexts derived from Lemma 7.1, which satisfies:
  - Σ_i^* is a uniform QPT algorithm, if Σ_i is a uniform QPT algorithm.
  - Σ_i^* satisfies correctness.
  - Σ_i^* satisfies (one-time) IND-CPA security and (one-time) unclonable IND-CPA security if Σ_i is uniform QPT algorithm and satisfies correctness, (one-time) IND-CPA security, and (one-time) unclonable IND-CPA security.

Construction of Robust (One-Time) Unclonable Secret-Key Encryption. A robust combiner for (one-time) unclonable secret-key encryption with $\ell(\lambda)$-bit plaintexts is a deterministic classical polynomial-time Turing machine that takes as input 1^n and $\{\Sigma_i\}_{i\in[n]}$, and outputs the following set of algorithms $\Sigma = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$:

$\mathsf{KeyGen}(1^\lambda)$**:**
- For all $i \in [n]$, run $\mathsf{sk}_i^* \leftarrow \mathsf{KeyGen}_i^*(1^\lambda)$.
- Output $\mathsf{sk} := \{\mathsf{sk}_i^*\}_{i\in[n]}$.

$\mathsf{Enc}(1^\lambda, \mathsf{sk}, m)$**:**
- For all $i \in [n]$, sample $r_i \leftarrow \{0,1\}^{\ell(\lambda)}$ promised that $\sum_{i\in[n]} r_i = m$, where the $\ell(\lambda)$ is the length of plaintext m.
- For all $i \in [n]$, run $\mathsf{CT}_i^* \leftarrow \mathsf{Enc}_i^*(1^\lambda, \mathsf{sk}_i^*, r_i)$ for all $i \in [n]$.
- Output $\mathsf{CT} := \{\mathsf{CT}_i^*\}_{i\in[n]}$.

$\mathsf{Dec}(1^\lambda, \mathsf{sk}, \mathsf{CT})$**:**
- Run $r_i^* \leftarrow \mathsf{Dec}_i^*(1^\lambda, \mathsf{sk}_i^*, \mathsf{CT}_i^*)$ for all $i \in [n]$.
- Output $\sum_{i\in[n]} r_i^*$.

Theorem 7.1 follows from the following Lemmata 7.2 to 7.5.

Lemma 7.2. *If all of $\{\Sigma_i\}_{i\in[n]}$ satisfies efficiency, Σ satisfies efficiency.*

Lemma 7.3. *Σ satisfies correctness.*

Lemma 7.4. *If all of $\{\Sigma_i\}_{i\in[n]}$ satisfies efficiency and one of $\{\Sigma_i\}_{i\in[n]}$, satisfies both correctness and (one-time) IND-CPA security, then Σ satisfies (one-time) IND-CPA security.*

Lemma 7.5. *If all of $\{\Sigma_i\}_{i\in[n]}$ satisfies efficiency and one of $\{\Sigma_i\}_{i\in[n]}$, satisfies both correctness and (one-time) unclonable IND-CPA security, then Σ satisfies (one-time) unclonable IND-CPA security.*

Lemmata 7.2 and 7.3 trivially follows, and thus we omit the proof. The proof of Lemmata 7.4 and 7.5 follow from the standard hybrid argument, and thus we omit the proof. □

7.1 Universal Constructions

Definition 7.2. *We say that a set of uniform QPT algorithms* $\Sigma_{\mathsf{Univ}} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *is a universal construction of (one-time) unclonable SKE (resp. PKE) if* Σ_{Univ} *is (one-time) unclonable SKE (resp. PKE) as long as there exists (one-time) unclonable SKE (resp. PKE).*

Theorem 7.2. *There exists a universal construction of (one-time) unclonable SKE and unclonable PKE.*

We omit the security proof. For details, please see the full version.

8 Universal Plaintext Extension for Unclonable Encryption

In this section, we prove the following Theorem 8.1.

Theorem 8.1. *Assume that there exists a decomposable quantum randomized encoding and one-time unclonable SKE* $\Sigma_{\mathsf{unclone}} = \mathsf{Unclone}.(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *where the size of the quantum circuit* $\mathsf{Unclone}.\mathsf{Dec}(1^\lambda, \cdot, \cdot)$ *is* $\ell(\lambda)$*. Then, for all polynomial* n*, there exists a polynomial* p *which depends on the polynomial* n *and* ℓ *and a set of uniform QPT algorithms* $\Sigma = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *which depends on the polynomial* p *such that* Σ *is a one-time unclonable secret-key encryption for* $n(\lambda)$*-bit plaintexts.*

Remark 8.1. Our construction is universal construction for one-time unclonable SKE in the sense that our construction does not depend on the single-bit scheme $\Sigma_{\mathsf{unclone}}$ that is assumed to exist except for the size of the decryption circuit of $\Sigma_{\mathsf{unclone}}$.

As corollaries, we obtain Corollaries 8.1 and 8.2.

Corollary 8.1. *For all polynomial* n*, there exists a set of uniform QPT algorithms* $\Sigma = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *such that* Σ *is unclonable secret-key encryption for* $n(\lambda)$*-bit plaintexts if there exists unclonable secret-key encryption for single-bit plaintexts.*

Corollary 8.2. *For all polynomial* n*, there exists a set of uniform QPT algorithms* $\Sigma = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *such that* Σ *is unclonable public-key encryption for* $n(\lambda)$*-bit plaintexts if there exists unclonable public-key encryption for single-bit plaintexts.*

Proof of Corollary 8.2. We give a rough sketch of the proof of Corollary 8.2. Note that, in the same way, we can prove Corollary 8.1.

We can construct PKE with quantum ciphertexts and one-time unclonable SKE with single-bit plaintexts from unclonable PKE for single-bit plaintexts. We can construct decomposable quantum randomized encoding from PKE with

quantum ciphertexts. Furthermore, from Theorem 8.1, we can construct one-time unclonable SKE with $n(\lambda)$-bit plaintexts from decomposable quantum randomized encoding and one-time unclonable SKE with single-bit plaintexts.

On the other hand, we can construct receiver non-committing encryption with quantum ciphertexts from PKE with quantum ciphertexts. By combining the receiver non-committing encryption with quantum ciphertexts and one-time unclonable SKE with $n(\lambda)$-bit plaintexts, we obtain unclonable PKE with $n(\lambda)$-bit plaintexts (For the detail, please see the full version.). □

Proof of Theorem 8.1. First, let us describe notations and observations.

Notations and Observations.

- Let $C_{\lambda,p}[m]$ be a quantum circuit of size $p(\lambda)$ with λ-qubit quantum inputs and λ-bit classical inputs such that it outputs m for any inputs, where p is a polynomial which we specify later.
- Let $\Sigma_{\mathsf{RE}} := \mathsf{RE}.(\mathsf{Enc}, \mathsf{Dec})$ be a decomposable quantum randomized encoding. Given quantum circuit C and n_1-length quantum input and n_2-length classical input $\mathbf{q}$ and x, the encoding $\widehat{C}(\mathbf{q}, x)$ can be separated as follows:

$$\widehat{C}(\mathbf{q}, x, r, e) = (\widehat{C}_{\mathsf{off}}, \widehat{C}_1, \cdots, \widehat{C}_{n_1+n_2})(\mathbf{q}, x, r, e),$$

  where r is uniformly ransom string and e is some quantum state. From decomposability, $\widehat{C}_{\mathsf{off}}$ acts only on r and e, and $\widehat{C_i}$ acts only on $\mathbf{q_i}, r$ and e for $i \in [n_1]$, and $\widehat{C_i}$ acts only on x_i and r for $i \in \{n_1 + 1, \cdots, n_1 + n_2\}$. For any quantum circuit C, we write $\mathsf{lab}[i, x_i] = \widehat{C_i}(x_i, r_i)$ and $\mathsf{lab}[i, \mathbf{q}_i] = \widehat{C_i}(\mathbf{q}_i, r, e)$.

Construction. We give a construction of one-time unclonable secret-key encryption $\Sigma := (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ with $n(\lambda)$-bit plaintexts by using decomposable quantum randomized encoding. In the construction, we only use decomposable quantum randomized encoding. The construction is secure as long as the underlying decomposable quantum randomized encoding is secure and there exists one-time unclonable secret-key encryption for single-bit plaintexts.

$\mathsf{KeyGen}(1^\lambda)$**:**

- Sample $x \leftarrow \{0,1\}^\lambda$.
- Sample $R[i] \leftarrow \{0,1\}^{\ell(\lambda)}$ for all $i \in [\lambda]$.
- Output $\mathsf{sk} := \big(x, \{R[i]\}_{i\in[\lambda]}\big)$.

$\mathsf{Enc}(1^\lambda, \mathsf{sk}, m)$**:**

- Parse $\mathsf{sk} = \big(x, \{R[i]\}_{i\in[\lambda]}\big)$.
- Prepare the quantum circuit $C_{\lambda,p}[m]$ that outputs m for any inputs.
- Compute $\widehat{C_{\lambda,p}}[m]_{\mathsf{off}}$.
- Compute $\{\mathsf{lab}[i,0]\}_{i\in[\lambda]}$, and $\{\mathsf{lab}[i,b]\}_{i\in\{\lambda+1,\cdots,2\lambda\},b\in\{0,1\}}$.
- Sample $S[i] \leftarrow \{0,1\}^{\ell(\lambda)}$ for all $i \in [\lambda]$.
- Compute $\mathsf{Lab.CT}[i+\lambda, x[i]] := R[i] + \mathsf{lab}[i+\lambda, x[i]]$ and $\mathsf{Lab.CT}[i+\lambda, 1-x[i]] := S[i] + \mathsf{lab}[i+\lambda, 1-x[i]]$ for all $i \in [\lambda]$.

– Output

$$\mathsf{CT} := \left(\widehat{C_{\lambda,p}}[m]_{\mathsf{off}},\ \{\mathsf{lab}[i,0]\}_{i\in[\lambda]}, \{\mathsf{Lab.CT}[i,b]\}_{i\in\{\lambda+1,\cdots,2\lambda\},b\in\{0,1\}}\right).$$

$\mathsf{Dec}(1^\lambda, \mathsf{sk}, \mathsf{CT})$**:**

– Parse $\mathsf{sk} = \left(x, \{R[i]\}_{i\in[\lambda]}\right)$ and

$$\mathsf{CT} = \left(\widehat{C_{\lambda,p}}[m]_{\mathsf{off}}, \{\mathsf{lab}[i,0]\}_{i\in[\lambda]}, \{\mathsf{Lab.CT}[i,b]\}_{i\in\{\lambda+1,\cdots,2\lambda\},b\in\{0,1\}}\right).$$

– Compute $\mathsf{lab}[i+\lambda, x[i]] := \mathsf{Lab.CT}[i+\lambda, x[i]] + R[i]$ for all $i \in [\lambda]$.

– Compute

$$\mathsf{RE.Dec}\left(\widehat{C_{\lambda,p}}[m]_{\mathsf{off}}, \{\mathsf{lab}[i,0]\}_{i\in[\lambda]}, \{\mathsf{lab}[i,x[i]]\}_{i\in\{\lambda+1,\cdots,2\lambda\}}\right)$$

and outputs its output.

Lemma 8.1. *Σ satisfies efficiency if Σ_{RE} is decomposable quantum randomized encoding.*

Lemma 8.2. *Σ satisfies correctness if Σ_{RE} is decomposable quantum randomized encoding.*

Lemma 8.3. *If Σ_{RE} is decomposable quantum randomized encoding and there exists one-time unclonable secret-key encryption with single-bit plaintexts, Σ satisfies one-time IND-CPA security for some polynomial p.*

Lemma 8.4. *If Σ_{RE} is decomposable quantum randomized encoding and there exists one-time unclonable secret-key encryption with single-bit plaintexts, Σ satisfies one-time unclonable IND-CPA security for some polynomial p.*

Lemma 8.1 straightforwardly follows. We can see that Lemma 8.2 holds as follows. First, if $\mathsf{sk} \leftarrow \mathsf{KeyGen}(1^\lambda)$ and $\mathsf{CT} \leftarrow \mathsf{Enc}(\mathsf{sk}, m)$, $\mathsf{Dec}(\mathsf{sk}, \mathsf{CT})$ outputs the output of $C_{\lambda,p}[m](0^\lambda, x)$. From the definition of $C_{\lambda,p}[m]$, $C_{\lambda,p}[m](0^\lambda, x)$ outputs m for all x.

The proof of Lemma 8.3 is the same as Lemma 8.4, and thus we skip the proof.

Proof of Lemma 8.4. By a standard argument, we can show the following Proposition 8.1.

Proposition 8.1. *If there exists one-time unclonable secret-key encryption for single-bit plaintexts, then there exists a one-time unclonable secret-key encryption for single-bit plaintexts scheme $\Sigma_{\mathsf{unclone}} = \mathsf{Unclone}.(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ such that the following properties are satisfied:*

1. *$\Sigma_{\mathsf{unclone}}$ satisfies perfect correctness.*
2. *For all security parameters $\lambda \in \mathbb{N}$ and $b \in \{0,1\}$, we have $|\mathsf{sk}_\lambda| = |\mathsf{CT}_{\lambda,b}| = \lambda$, where $\mathsf{sk}_\lambda \leftarrow \mathsf{Unclone.KeyGen}(1^\lambda)$ and $\mathsf{CT}_{\lambda,b} \leftarrow \mathsf{Unclone.Enc}(1^\lambda, \mathsf{sk}_\lambda, b)$.*

3. *For all security parameters* λ, $\mathsf{Unclone.KeyGen}(1^\lambda)$ *uniformly randomly samples* sk_λ.

We can show the Proposition 8.1 by correctness amplification and standard padding argument. We omit the proof. For details, please see the full version. We define$D_\lambda[m_0, m_1]$ as a quantum circuit that takes as input λ-qubit quantum inputs ρ and λ-bit classical bits x, runs the quantum circuit $b \leftarrow \mathsf{Unclone.Dec}(1^\lambda, x, \rho)$, and outputs m_b. Now, we define p as a polynomial large enough to run the circuit $D_\lambda[m_0, m_1]$.

We describe the sequence of hybrids against the adversary $(\mathcal{A}, \mathcal{B}, \mathcal{C})$.

Hyb_0**:** This is the original one-time unclonable IND-CPA security experiment.

1. The challenger samples $b \leftarrow \{0,1\}$.
2. The challenger samples $x \leftarrow \{0,1\}^\lambda$ and $R[i] \leftarrow \{0,1\}^{\ell(\lambda)}$ for all $i \in [\lambda]$.
3. $\mathcal{A}$ sends (m_0, m_1) to the challenger.
4. The challenger computes $\widehat{C_{\lambda,p}}[m_b]_{\mathsf{off}}$, $\{\mathsf{lab}[i,0]\}_{i\in[\lambda]}$, and $\{\mathsf{lab}[i,\beta]\}_{i\in\{\lambda+1,\cdots,2\lambda\},\beta\in\{0,1\}}$.
5. The challenger samples $S[i] \leftarrow \{0,1\}^{\ell(\lambda)}$ for all $i \in [\lambda]$, and computes

$$\mathsf{Lab.CT}[i+\lambda, x[i]] := R[i] + \mathsf{lab}[i+\lambda, x[i]]$$
$$\mathsf{Lab.CT}[i+\lambda, 1-x[i]] := S[i] + \mathsf{lab}[i+\lambda, 1-x[i]]$$

   for all $i \in [\lambda]$.
6. The challenger sends

$$\mathsf{CT} := \left(\widehat{C_{\lambda,p}}[m]_{\mathsf{off}}, \{\mathsf{lab}[i,0]\}_{i\in[\lambda]}, \{\mathsf{Lab.CT}[i,\beta]\}_{i\in\{\lambda+1,\cdots,2\lambda\},\beta\in\{0,1\}}\right).$$

   to $\mathcal{A}$.
7. $\mathcal{A}$ produces $\rho_{\mathcal{B},\mathcal{C}}$ and sends the corresponding registers to $\mathcal{B}$ and $\mathcal{C}$.
8. $\mathcal{B}$ and $\mathcal{C}$ receives $\left(x, \{R[i]\}_{i\in[\lambda]}\right)$, and outputs $b_\mathcal{B}$ and $b_\mathcal{C}$.
9. The experiment outputs 1 if $b_\mathcal{B} = b_\mathcal{C} = b$, and otherwise 0.

Hyb_1**:**

1. The challenger samples $b \leftarrow \{0,1\}$.
2. The challenger samples $x \leftarrow \{0,1\}^\lambda$ and $R[i] \leftarrow \{0,1\}^{\ell(\lambda)}$ for all $i \in [\lambda]$.
3. The adversary $\mathcal{A}$ sends (m_0, m_1) to the challenger.
4. The challenger computes $\mathsf{unclone.CT}_b \leftarrow \mathsf{Unclone.Enc}(1^\lambda, x, b)$, where $\mathsf{unclone.CT}_b$ is the λ-length quantum states.
5. The challenger computes $\widehat{D_\lambda}[m_0, m_1]_{\mathsf{off}}$, $\{\mathsf{lab}[i, \mathsf{unclone.CT}_b[i]]\}_{i\in[\lambda]}$, and $\{\mathsf{lab}[i,\beta]\}_{i\in\{\lambda+1,\cdots,2\lambda\},\beta\in\{0,1\}}$.
6. The challenger samples $S[i] \leftarrow \{0,1\}^{\ell(\lambda)}$ for all $i \in [\lambda]$, and computes

$$\mathsf{Lab.CT}[i+\lambda, x[i]] := R[i] + \mathsf{lab}[i+\lambda, x[i]]$$
$$\mathsf{Lab.CT}[i+\lambda, 1-x[i]] := S[i] + \mathsf{lab}[i+\lambda, 1-x[i]]$$

   for all $i \in [\lambda]$.

7. The challenger sends
$$\mathsf{CT} := \big(\widehat{D_\lambda}[m_0, m_1]_{\mathsf{off}}, \{\mathsf{lab}[i, \mathsf{unclone.CT}_b[i]]\}_{i\in[\lambda]}, \{\mathsf{Lab.CT}[i,\beta]\}_{i\in\{\lambda+1,\cdots,2\lambda\},\beta\in\{0,1\}}\big)$$
to $\mathcal{A}$.
8. $\mathcal{A}$ produces $\rho_{\mathcal{B},\mathcal{C}}$ and sends the corresponding registers to $\mathcal{B}$ and $\mathcal{C}$.
9. $\mathcal{B}$ and $\mathcal{C}$ receives $\big(x, \{R[i]\}_{i\in[\lambda]}\big)$, and outputs $b_{\mathcal{B}}$ and $b_{\mathcal{C}}$.
10. The experiment outputs 1 if $b_{\mathcal{B}} = b_{\mathcal{C}} = b$, and otherwise 0.

Lemma 8.4 follows from the following Propositions 8.2 and 8.3.

Proposition 8.2. *If Σ_{RE} is decomposable quantum randomized encoding, then*
$$|\Pr[\mathsf{Hyb}_0 = 1] - \Pr[\mathsf{Hyb}_1 = 1]| \leq \mathsf{negl}(\lambda).$$

The challenger in Hyb_0 uses $\Big(\widehat{C_{\lambda,p}}[m_b]_{\mathsf{off}}, \{\mathsf{lab}[i,0]\}_{i\in[\lambda]}, \{\mathsf{Lab.CT}[i,\beta]\}_{i\in\{\lambda+1,\cdots,2\lambda\},\beta\in\{0,1\}}\Big)$ as a quantum ciphertext. In Hyb_1, as a quantum ciphertext, the challenger uses $\Big(\widehat{D_\lambda}[m_0,m_1]_{\mathsf{off}}, \{\mathsf{lab}[i,\mathsf{unclone.CT}_b[i]]\}_{i\in[\lambda]}, \{\mathsf{Lab.CT}[i,\beta]\}_{i\in\{\lambda+1,\cdots,2\lambda\},\beta\in\{0,1\}}\Big)$ instead. Furthermore, we have $C_{\lambda,p}[m_b](0^\lambda, x) = D_\lambda[m_0,m_1](\mathsf{unclone.CT}_b, x) = m_b$. Therefore, by Proposition 3.1, if we can distinguish Hyb_0 from Hyb_1, then we can also break the security of decomposable quantum randomized encoding. For details, please see the full version.

Proposition 8.3. *If there exists a one-time unclonable secret-key encryption $\Sigma_{\mathsf{Unclone}}$ with single-bit plaintexts, then*
$$|\Pr[\mathsf{Hyb}_1 = 1]| \leq \frac{1}{2} + \mathsf{negl}(\lambda).$$

We give a rough sketch of Proposition 8.3. The adversary of $(\widetilde{\mathcal{A}}, \widetilde{\mathcal{B}}, \widetilde{\mathcal{C}})$ can break the security of arbitrary unclonable SKE $\Sigma_{\mathsf{Unclone}}$ with single-bit plaintexts by using $(\mathcal{A},\mathcal{B},\mathcal{C})$, which breaks the security of Hyb_1. By receiving the $\mathsf{unclone.CT}_b$ from the challenger of $\Sigma_{\mathsf{Unclone}}$, $\widetilde{\mathcal{A}}$ generates $\Big(\widehat{D_\lambda}[m_0,m_1]_{\mathsf{off}}, \{\mathsf{lab}[i,\mathsf{unclone.CT}_b[i]]\}_{i\in[\lambda]}, \{\mathsf{Lab.CT}[i,\beta]\}_{i\in\{\lambda+1,\cdots,2\lambda\},\beta\in\{0,1\}}\Big)$, runs $\mathcal{A}$ on it, and passes the corresponding register to $\widetilde{\mathcal{B}}$ and $\widetilde{\mathcal{C}}$. $\widetilde{\mathcal{B}}$ (resp. $\widetilde{\mathcal{C}}$) runs $\mathcal{B}$ (resp. $\mathcal{C}$) on the corresponding register, and outputs its output. From the construction of $\mathcal{A}$, $\mathcal{A}$ perfectly simulates the challenger of Hyb_1. Therefore, $(\mathcal{A},\mathcal{B},\mathcal{C})$ can break the security of arbitrary unclonable SKE $\Sigma_{\mathsf{Unclone}}$ with single-bit plaintexts. □

□

Acknowledgements. We thank Mark Zhandry for pointing out a bug in our robust combiner for public key quantum money in the previous version. TH is supported by JSPS research fellowship and by JSPS KAKENHI No. JP22J21864.

References

Aar18. Aaronson, S.: Shadow tomography of quantum states. In: Diakonikolas, I., Kempe, D., Henzinger, M. (eds.) 50th ACM STOC, pp. 325–338. ACM Press (2018)

ABJ+19. Ananth, P., Badrinarayanan, S., Jain, A., Manohar, N., Sahai, A.: From FE combiners to secure MPC and back. In: Hofheinz, D., Rosen, A. (eds.) TCC 2019. Part I, volume 11891 of LNCS, pp. 199–228. Springer, Heidelberg (2019)

AC12. Aaronson, S., Christiano, P.: Quantum money from hidden subspaces. In: STOC, pp. 41–60. ACM (2012)

AGQY22. Ananth, P., Gulati, A., Qian, L., Yuen, H.: Pseudorandom (function-like) quantum state generators: new definitions and applications. In: Kiltz, E., Vaikuntanathan, V. (eds.) TCC 2022, Part I, pp. 237–265. Springer, Cham (2022)

AJN+16. Ananth, P., Jain, A., Naor, M., Sahai, A., Yogev, E.: Universal constructions and robust combiners for indistinguishability obfuscation and witness encryption. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016. LNCS, Part II, vol. 9815, pp. 491–520. Springer, Heidelberg (2016)

AJS17. Ananth, P., Jain, A., Sahai, A.: Robust transforming combiners from indistinguishability obfuscation to functional encryption. In: EUROCRYPT (1), pp. 91–121. Springer (2017)

AK21. Ananth, P., Kaleoglu, F.: Unclonable encryption, revisited. In: Nissim, K., Waters, B. (eds.) Theory of Cryptography, pp. 299–329. Springer, Cham (2021)

AKL+22. Ananth, P., Kaleoglu, F., Li, X., Liu, Q., Zhandry, M.: On the feasibility of unclonable encryption, and more. In: Dodis, Y., Shrimpton, T. (eds.) Advances in Cryptology – CRYPTO 2022, pp. 212–241. Springer, Cham (2022)

AQY22. Ananth, P., Qian, L., Yuen, H.: Cryptography from pseudorandom quantum states. In: Dodis, Y., Shrimpton, T. (eds.) Advances in Cryptology – CRYPTO 2022, pp. 208–236. Springer, Cham (2022)

BB84. Bennett, C.H., Brassard, G.: Quantum cryptography: public key distribution and coin tossing. In: Proceedings of IEEE International Conference on Computers, Systems, and Signal Processing, p. 175 (1984)

BCKM21. Bartusek, J., Coladangelo, A., Khurana, D., Ma, F.: On the round complexity of secure quantum computation. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021. LNCS, Part I, vol. 12825, pp. 406–435. Springer, Heidelberg (2021)

BCQ23. Brakerski, Z., Canetti, R., Qian, L.: On the computational hardness needed for quantum cryptography. In: Kalai, Y.T. (ed.) 14th Innovations in Theoretical Computer Science Conference, ITCS 2023, 10–13 January 2023, MIT, Cambridge. LIPIcs, vol. 251, pp. 24:1–24:21. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2023)

BL20. Broadbent, A., Lord, S.: Uncloneable quantum encryption via oracles. In: Flammia, S.T. (ed.) 15th Conference on the Theory of Quantum Computation, Communication and Cryptography, TQC 2020, 9–12 June 2020, Riga. LIPIcs, vol. 158, pp. 4:1–4:22. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2020)

BY22. Brakerski, Z., Yuen, H.: Quantum garbled circuits. In: Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2022, pp. 804–817. Association for Computing Machinery, New York (2022)
CHK05. Canetti, R., Halevi, S., Katz, J.: Adaptively-secure, non-interactive public-key encryption. In: Kilian, J. (ed.) TCC 2005. LNCS, vol. 3378, pp. 150–168. Springer, Heidelberg (2005)
CLS01. Crépeau, C., Légaré, F., Salvail, L.: How to convert the flavor of a quantum bit commitment. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 60–77. Springer, Heidelberg (2001)
CX22. Cao, S., Xue, R.: On constructing one-way quantum state generators, and more. Cryptology ePrint Archive, Paper 2022/1323 (2022). https://eprint.iacr.org/2022/1323
DH76. Diffie, W., Hellman, M.E.: New directions in cryptography. IEEE Trans. Inf. Theory **22**(6), 644–654 (1976)
DMS00. Dumais, P., Mayers, D., Salvail, L.: Perfectly concealing quantum bit commitment from any quantum one-way permutation. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 300–315. Springer, Heidelberg (2000)
ElG85. ElGamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Trans. Inf. Theory **31**, 469–472 (1985)
FGH+12. Farhi, E., Gosset, D., Hassidim, A., Lutomirski, A., Shor, P.W.: Quantum money from knots. In: Goldwasser, S. (ed.) ITCS 2012, pp. 276–289. ACM (2012)
GLSV21. Grilo, A.B., Lin, H., Song, F., Vaikuntanathan, V.: Oblivious transfer is in MiniQCrypt. In: Canteaut, A., Standaert, F.-X. (eds.) EUROCRYPT 2021. LNCS, Part II, vol. 12697, pp. 531–561. Springer, Heidelberg (2021)
GTK16. Goldwasser, S., Tauman Kalai, Y.: Cryptographic assumptions: a position paper. In: Kushilevitz, E., Malkin, T. (eds.) Theory of Cryptography, pp. 505–522. Springer, Heidelberg (2016)
Her05. Herzberg, A.: On tolerant cryptographic constructions. In: Menezes, A. (ed.) Topics in Cryptology – CT-RSA 2005, pp. 172–190. Springer, Heidelberg (2005)
HKN+05. Harnik, D., Kilian, J., Naor, M., Reingold, O., Rosen, A.: On robust combiners for oblivious transfer and other primitives. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 96–113. Springer, Heidelberg (2005)
HMNY21. Hiroka, T., Morimae, T., Nishimaki, R., Yamakawa, T.: Quantum encryption with certified deletion, revisited: public key, attribute-based, and classical communication. In: Tibouchi, M., Wang, H. (eds.) Advances in Cryptology – ASIACRYPT 2021, pp. 606–636. Springer, Cham (2021)
HMY23. Hhan, M., Morimae, T., Yamakawa, T.: From the hardness of detecting superpositions to cryptography: quantum public key encryption and commitments. In: Hazay, C., Stam, M. (eds.) Advances in Cryptology – EUROCRYPT 2023, pp. 639–667. Springer, Cham (2023)
JLS18. Ji, Z., Liu, Y.-K., Song, F.: Pseudorandom quantum states. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018. LNCS, Part III, vol. 10993, pp. 126–152. Springer, Heidelberg (2018)
JMS20. Jain, A., Manohar, N., Sahai, A.: Combiners for functional encryption, unconditionally. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020. LNCS, Part I, vol. 12105, pp. 141–168. Springer, Heidelberg (2020)
Kan18. Kane, D.M.: Quantum money from modular forms. arXiv preprint arXiv:1809.05925 (2018)

KNTY19. Kitagawa, F., Nishimaki, R., Tanaka, K., Yamakawa, T.: Adaptively secure and succinct functional encryption: improving security and efficiency, simultaneously. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019. LNCS, Part III, vol. 11694, pp. 521–551. Springer, Heidelberg (2019)

KQST23. Kretschmer, W., Qian, L., Sinha, M., Tal, A.: Quantum cryptography in algorithmica. In: Proceedings of the 55th Annual ACM Symposium on Theory of Computing, STOC 2023, pp. 1589–1602. Association for Computing Machinery, New York (2023)

Kre21. Kretschmer, W.: Quantum pseudorandomness and classical complexity. In: Hsieh, M.-H. (ed.) 16th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2021). Leibniz International Proceedings in Informatics (LIPIcs), vol. 197, pp. 2:1–2:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl (2021)

KSS22. Kane, D.M., Sharif, S., Silverberg, A.: Quantum money from quaternion algebras (2022)

KT23. Khurana, D., Tomer, K.: Commitments from quantum one-wayness (2023)

LC97. Lo, H.-K., Chau, H.F.: Is quantum bit commitment really possible? Phys. Rev. Lett. **78**, 3410–3413 (1997)

Lev85. Levin, L.A.: One-way functions and pseudorandom generators. In: 17th ACM STOC, pp. 363–365. ACM Press (1985)

LMZ23. Liu, J., Montgomery, H., Zhandry, M.: Another round of breaking and making quantum money: In: Hazay, C., Stam, M. (eds.) Advances in Cryptology – EUROCRYPT 2023, pp. 611–638. Springer, , Cham (2023)

May97. Mayers, D.: Unconditionally secure quantum bit commitment is impossible. Phys. Rev. Lett. **78**, 3414–3417 (1997)

MY22a. Morimae, T., Yamakawa, T.: One-wayness in quantum cryptography. Cryptology ePrint Archive, Paper 2022/1336 (2022). https://eprint.iacr.org/2022/1336

MY22b. Morimae, T., Yamakawa, T.: Quantum commitments and signatures without one-way functions. In: Dodis, Y., Shrimpton, T. (eds.) Advances in Cryptology – CRYPTO 2022, pp. 269–295. Springer, Cham (2022)

Reg05. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: Gabow, H.N., Fagin, R. (eds.) 37th ACM STOC, pp. 84–93. ACM Press (2005)

WW23. Waters, B., Wichs, D.: Universal amplification of KDM security: from 1-key circular to multi-key KDM. Cryptology ePrint Archive, Paper 2023/1058 (2023). https://eprint.iacr.org/2023/1058

Yan22. Yan, J.: General properties of quantum bit commitments (extended abstract). In: Agrawal, S., Lin, D. (eds.) Advances in Cryptology – ASIACRYPT 2022, pp. 628–657. Springer, Cham (2022)

Zha19. Zhandry, M.: Quantum lightning never strikes the same state twice. In: Ishai, Y., Rijmen, V. (eds.) EUROCRYPT 2019. Part III, volume 11478 of LNCS, pp. 408–438. Springer, Heidelberg (2019)

Zha23a. Zhandry, M.: Quantum minimalism (talk) (2023). https://www.youtube.com/watch?v=7cqnrASfjco&ab_channel=SimonsInstitute

Zha23b. Zhandry, M.: Quantum money from abelian group actions. IACR Cryptol. ePrint Arch. **2023**, 1097 (2023)

Unbounded Leakage-Resilience and Intrusion-Detection in a Quantum World

Alper Çakan[1(✉)], Vipul Goyal[1,2], Chen-Da Liu-Zhang[3], and João Ribeiro[4]

[1] Carnegie Mellon University, Pittsburgh, PA, USA
acakan@cs.cmu.edu
[2] NTT Research, Sunnyvale, CA, USA
vipul@vipulgoyal.org
[3] Lucerne University of Applied Sciences and Arts & Web3 Foundation, Lucerne, Switzerland
chen-da.liuzhang@hslu.ch
[4] Instituto de Telecomunicações and Departamento de Matemática, Instituto Superior Técnico, Universidade de Lisboa, Lisboa, Portugal
jribeiro@tecnico.ulisboa.pt

Abstract. Can an adversary hack into our system and steal sensitive data such as cryptographic keys? This question is as old as the Internet and significant effort has been spent on designing mechanisms to prevent and detect hacking attacks. Once quantum computers arrive, will the situation remain the same or can we hope to live in a better world?

We first consider ubiquitous side-channel attacks, which aim to leak side information on secret system components, studied in the *leakage-resilient* cryptography literature. Classical leakage-resilient cryptography must necessarily impose restrictions on the type of leakage one aims to protect against, such as the popular *bounded leakage* model. Although such leakage bounds are necessary, many real-world side-channel attacks cannot be captured by bounded leakage. In this work, we design cryptographic schemes that provide guarantees against *arbitrary* side-channel attacks:

- Using techniques from unclonable quantum cryptography, we design several basic leakage-resilient primitives, such as public- and private-key encryption, pseudorandom functions, digital signatures and quantum money schemes which remain secure under *unbounded* adaptive classical leakage over *unbounded* number of rounds.
- What if the adversary simply breaks into our system to steal our secret keys, rather than mounting only a side-channel attack? What if the adversary can even tamper with the data arbitrarily, for example to cover its tracks? We initiate the study of *intrusion-detection* in the quantum setting, where one would like to detect if security has been compromised even in the face of such attacks. We design cryptographic schemes supporting intrusion-detection for a host of primitives such as public- and private-key encryption, digital signature,

E. Boyle and M. Mahmoody (Eds.): TCC 2024, LNCS 15365, pp. 159–191, 2025.
https://doi.org/10.1007/978-3-031-78017-2_6

functional encryption, program obfuscation and software protection. Our schemes are based on techniques from cryptography with secure key leasing and certified deletion.

Keywords: Quantum cryptography · Leakage resilience

1 Introduction

Securely storing sensitive data is a central problem in computer security. This could mean that we would like to make any intrusion into the system harder to realize, as well as detect if an intrusion did occur. This is a notoriously hard problem with significant resources spent on preventing and mitigating such attacks. Indeed, in the classical setting all information can theoretically be copied, and so we can only rely on heuristic countermeasures for such attacks. We study this question in the quantum setting and show that the quantum world offers security guarantees that are classically impossible.

We first consider side-channel (also known as *leakage*) attacks. Real-world implementations of cryptographic schemes are often vulnerable to side-channel attacks, which allow an adversary to obtain side information from secret components such as a secret key. This can be achieved, for example, by measuring the time elapsed or the electromagnetic radiation emitted during computations—such simple practical attacks stretch back some decades [4,19,22] and have proven catastrophic for textbook versions of several well known schemes. As a response to this, *leakage-resilient cryptography*, the study of cryptographic schemes resilient against many types of side-channel attacks, has received significant interest. The survey of Kalai and Reyzin [17] is an excellent source for many of the developments in this area.

Arguably the most well-studied leakage model is that of *bounded leakage*. In this model, it is assumed that the adversary may not leak more than ℓ bits of leakage from a secret component, where ℓ is some leakage bound that is smaller than the secret length k. For example, in the setting of encryption, with a secret key $\mathsf{sk} \in \{0,1\}^k$, the adversary chooses an arbitrary function $f : \{0,1\}^k \to \{0,1\}^\ell$, where $\ell < k$ represents the leakage bound, and learns the bounded leakage $f(\mathsf{sk})$.

Is a Leakage Bound Justified? Generally, the justification for a leakage bound is that in the absence of one, the adversary can just leak the whole secret and no security guarantees are possible. While the study of bounded leakage-resilient cryptography has given rise to a beautiful and highly successful area of research, it is quite often the case that real world side channels attacks do not adhere to any a priori bounded leakage limit [13]. Moreover, even choosing a leakage bound entails predicting adversarial capabilities, and these predictions may be wildly incorrect.

Leakage-Resilience in a Quantum World. What if we store the secret being leaked on as a quantum state? Quantum information behaves in a fundamentally different way compared to its classical counterpart. While classical schemes can only tolerate a bounded amount of leakage, the same may not be true for quantum schemes. This raises the following tantalizing question:

> Is it possible to design cryptographic schemes based on the laws of quantum mechanics which can tolerate arbitrary unbounded leakage?

We answer the above question in the affirmative by proposing a host of cryptographic schemes resilient to *unbounded classical leakage*. Based on the literature on side-channel attacks, our view is that leakage is the result of *observations* of quantities such as electromagnetic radiation, power consumption, time elapsed, and temperature fluctuations. Hence, leakage or side-channel information is indeed classical information. In fact, we further define the notion of *LOCC (local operations and classical communication) leakage* where the adversary can *adaptively* obtain unbounded classical leakage over an *unbounded* number of rounds, and show that our schemes are also resilient against such attacks. Therefore, we believe that *our schemes can even be seen as leakage-proof rather than just leakage-resilient.*

We design a host of cryptographic schemes such as public- and private-key encryption schemes, digital signatures, pseudorandom functions and quantum money schemes which are resilient to LOCC leakage.

Intrusion-Detection in a Quantum World. What if an adversary, rather than mounting a side-channel attack, simply breaks into our system? In this case, the adversary can just obtain our whole secret state and, analogously to the classical setting, all bets are now off. Even worse, what if the adversary can even tamper with the stored data in an arbitrary manner (for example, to cover its tracks)? Can we achieve meaningful security guarantees in the face of such attacks?

We refer to the above setting as *intrusion-detection* in the quantum world. While intrusion-detection is fundamentally impossible in the classical setting (since an adversary may just clone secret system components without causing any changes to the system's state), it has nonetheless been widely studied in practice and is considered a highly desirable security goal. For example, tamper-proof audit logs have been extensively studied which, under certain assumptions, can detect if a machine has been broken into [6,25,26].

Surprisingly, based on principles of quantum mechanics, we are able to design many primitives supporting intrusion-detection. More precisely, our schemes provide the following guarantee. Suppose that an adversary was able to arbitrarily act on the secret, tampering it and obtaining sufficient information to break the security of the primitive (e.g., break indistinguishability in the case of public-key encryption). Then, a procedure called TestIntrusion outputs INTRUSION with overwhelming probability, indicating that an attack occurred and security has been compromised. TestIntrusion takes as input the residual secret (e.g., the

residual secret key in the case of public-key encryption), and *a classical public verification key.*[1]

We require that if the procedure TestIntrusion outputs NO INTRUSION, then, with overwhelming probability, *either there has been no attack, or any possible attack was not successful in breaking the scheme's security!* All of our results in this direction are obtained via a connection to cryptography with secure key leasing [8–10] (also called cryptography with certified deletion).

1.1 Our Results

LOCC Leakage-Resilience. We design schemes for public-key encryption (PKE), signatures, and pseudorandom functions (PRFs) that tolerate LOCC leakage against polynomial adversaries. For all tasks mentioned here, we consider an adversary that, over any polynomial number of rounds, adaptively specifies an unbounded measurement which is applied to the secret quantum key, and receives the (classical) measurement result and moves onto the next round of leakage (where the measurement will now be applied to the post-measurement state from the previous round). After the leakage phase is over, the adversary participates in the respective security game (e.g., in the case of PKE, it receives a challenge ciphertext and guesses the plaintext).

Theorem 1. *Assuming the existence of post-quantum sub-exponentially secure iO, one-way functions, and the quantum hardness of Learning with Errors (LWE) [23], there exist LOCC-leakage-resilient schemes for public-key encryption, digital signatures, and PRFs.*

We also introduce a new security notion for quantum money, called *unbounded leakage-resilient quantum banks*. In most quantum money schemes (e.g., [2,29]), the bank produces a banknote by sampling an unclonable quantum state through a public procedure, and signs its serial number using a classical signature scheme, where the secret key of the bank is simply the classical signing key sk of this classical signature scheme. However, we note that a leakage on the secret key sk of the bank would be catastrophic, as it would allow an adversary to produce any number of banknotes. Thus, we introduce the notion of *unbounded leakage-resilient quantum banks*. In this setting, the adversary obtains k banknotes along with unbounded classical leakage on the secret (quantum) key of the bank, and it still will not be able to create even a single extra banknote.

Theorem 2. *Assuming existence of quantum lightning [29], relative to a classical oracle, there exists a quantum bank scheme with unbounded classical leakage-resilience.*

[1] A copy of this public verification key could be stored offline or at multiple locations. Note that the notion of intrusion-detection is meaningless in the absence of such a public verification key, since in that case the adversary can simply swap our whole system, including the verification key, with a fresh instance of the scheme.

We similarly introduce the notion of chosen-message unbounded leakage-resilient signature schemes with quantum signatures[2].

Theorem 3. *Assuming existence of quantum lightning [29], relative to a classical oracle, there exists a signature scheme that satisfies chosen-message unbounded classical leakage-resilience.*

Relationship Between Leakage-Resilience and Unclonability. We are able to construct LOCC leakage-resilient schemes for various primitives using techniques from unclonable cryptography. This leads us to wonder whether a more general connection holds between unclonability (i.e., anti-piracy security) and LOCC leakage-resilience in general.

We first show that in some cryptographic settings unclonability implies (non-adaptive) unbounded leakage-resilience (Definition 4).

Theorem 4. *Let X be a {public-key encryption, digital signature, PRF} scheme that satisfies* random-challenge *anti-piracy security. Then, X also satisfies* random-challenge *non-adaptive unbounded classical leakage-resilience.*

While this might lead one to think that leakage-resilience in general is weaker than and implied by unclonability, we show that this is not the case. In fact, relative to a classical oracle, there exist schemes that satisfy unclonability (and thus non-adaptive unbounded leakage-resilience) but not even 1-round LOCC leakage-resilience (Definition 4).

Theorem 5 (informal). *Relative to a classical oracle, there exists a {public-key encryption, digital signature, PRF} scheme that satisfies anti-piracy security but does not satisfy* 1*-round LOCC leakage-resilience.*

We believe this supports our vision of LOCC leakage-resilience as a new research direction, since, aside from modelling side-channels attacks in the most general way and thus being an important security model, it is also not implied by existing security notions.

We also show that LOCC leakage-resilience does not imply unclonability either, which means that it might be possible to obtain leakage-resilience from weaker assumptions.

Theorem 6. *Assuming subexponentially secure $i\mathcal{O}$, one-way functions and qLWE, there exists a {public-key encryption, signature, PRF} scheme that satisfies LOCC leakage-resilience but there exists a cloning (i.e. piracy) adversary that wins the anti-piracy game with probability* 1*.*

Intrusion-Detection. As pointed out in the introduction, we cannot have resilience against the case where the adversary completely breaks into our system, since the adversary can leak the whole quantum secret. Therefore, in that

[2] Note that this notion is impossible with classical signatures since the adversary can simply leak a classical signature.

case, we aim to achieve intrusion-detection instead. More specifically, we design cryptographic primitives with an intrusion-detection algorithm which can detect whether *useful* information has been obtained on the secret key. These results are obtained by establishing a connection to cryptographic schemes with publicly verifiable secure leasing [9].

Theorem 7 (informal). *Suppose that there exists a* $X \in$ [public-key encryption, digital signature, PRF, functional encryption, obfuscation, software protection] *scheme with publicly verifiable secure leasing. Then, there exists an* X *scheme with intrusion-detection.*

We show that the other direction is also true: intrusion-detection implies publicly verifiable secure key-leasing with quantum certificates for the primitives listed above.

Based on the constructions from the secure leasing literature [9,20], we get the following corollaries of our theorem above.

Corollary 1. *Assuming post-quantum indistinguishability obfuscation and (injective) one-way functions, there exists a public-key encryption, PRF, functional encryption, differing-inputs obfuscation and software protection scheme with intrusion-detection.*

Corollary 2. *Assuming post-quantum subexponentially secure indistinguishability obfuscation, one-way functions, and quantum hardness of LWE, there exists a digital signature scheme with intrusion-detection.*

While $i\mathcal{O}$ is a strong assumption, we show that public-key quantum money is implied by intrusion-detection. Thus, our assumption can be considered unavoidable until a breakthrough is achieved in the construction of public-key quantum money, since it is a major open problem to construct it without $i\mathcal{O}$.

Theorem 8. *Suppose there exists a public-key encryption scheme with intrusion-detection. Then, there exists a public-key quantum money scheme.*

Our result can be similarly generalized to other primitives [15].

Constructions Based on BB84 States. We also construct (information-theoretic) leakage-resilient secret sharing schemes and (computational) private key encryption schemes that are secure in the unbounded leakage-resilience setting, by proving an LOCC leakage (+ bounded quantum leakage) resilience result for BB84 states.

Theorem 9 (informal). *Assuming existence of public-key encryption, there exist a secret key encryption scheme that satisfies LOCC leakage and constant rate bounded quantum leakage on its secret key.*

Theorem 10 (informal). *For any monotone access structure* Γ *with no singletons, there exists an information theoretically secure secret sharing scheme for* Γ *tolerating unentangled LOCC leakage and constant rate bounded quantum leakage from the shares. On the other hand, there cannot exists a secret sharing scheme that is even classically leakage-resilient against entangled leakage.*

We refer the reader to the full version [15] for formal statements, constructions and the proofs.

2 Technical Overview

A common theme across several of our results is that they are based on techniques from cryptography with unclonability (also called *copy-protection* or *anti-piracy* security and secure key leasing (i.e., keys with certified deletion). The connection between leakage-resilience and copy-protection is as follows. Suppose one can obtain classical leakage on a quantum secret satisfying copy-protection security, which is "functionally equivalent" to the secret itself (e.g., this leakage allows one to decrypt in case the quantum *secret* is a secret key of an encryption scheme). Since any classical information can be cloned, this gives us a way of essentially obtaining multiple states having the same functionality as the original quantum secret. Since we assumed the quantum secret was "unclonable", we arrive at a contradiction, which should yield leakage-resilience security. While this basic observation is indeed our starting point, this is not enough due to new challenges in the setting of leakage-resilience. We highlight some challenges, taking public-key encryption as our running example.

Challenge 1: Adversary's State Cannot be Cloned. In the context of public-key encryption, the LOCC leakage game consists first of an adversary $\mathcal{A}$ *adaptively* specifying a measurement to be applied to the key for multiple rounds. Then, the adversary will try to decrypt the challenge ciphertext with the information that it has obtained. The implicit assumption in the hypothetical reduction (from leakage-resilience to unclonability) from the previous paragraph is that there is a single measurement made by $\mathcal{A}$ which is sufficient to decrypt the challenge ciphertext. However, in general the LOCC leakage game will have multiple rounds of adaptive measurements, and the decryption of the challenge ciphertext will utilize the final (quantum) internal state of $\mathcal{A}$. In general, this state cannot be cloned even though descriptions of the measurements chosen by $\mathcal{A}$ are classical, and this holds true even if one started with multiple copies of the initial state of $\mathcal{A}$ as non-uniform advice. This is because $\mathcal{A}$ can choose the measurements in each round probabilistically and therefore the different copies of $\mathcal{A}$ could choose a different measurement to be applied to the key. However, only a single leakage measurement can be run since we only have a single copy of the secret key during the reduction, and the measurements can irreversibly change the state, resulting in an inability to answer multiple different measurement/leakage requests from the two copies of $\mathcal{A}$. This challenge becomes particularly interesting in the computational setting, as we discuss later.

Challenge 2: One Adversary vs. Two Adversaries. In PKE with unclonable secret keys [16], an adversary in possession of the decryption key $\mathsf{R}_{\mathsf{dec}}$ splits it across two adversaries in an arbitrary manner, and two challenge ciphertexts are then sent to these adversaries. Afterwards, we require that the probability that *both* adversaries can simultaneously correctly decrypt their ciphertexts is negligibly

close to 1/2. This means that the adversaries' baseline success probability in the unclonable decryption game is 1/2 (since one of the adversaries can simply keep the original decryption key $\mathsf{R}_{\mathsf{dec}}$ and correctly distinguish its challenge ciphertext with probability 1 and the other one can randomly guess with probability 1/2). On the other hand, the guarantee in leakage-resilient PKE requires that the probability of correctly decrypting *one* ciphertext given the leakage is negligibly close to 1/2. This means that the probability of correctly decrypting *two* ciphertexts, as in the unclonable decryption game, should be close to 1/4, instead of close to 1/2. This means that a direct reduction to the unclonable decryption game cannot be obtained. We run into a similar issue while considering unclonable PRFs as well.

The above challenges show that it is not possible to reduce leakage-resilience to unclonability in all settings. In particular, while in the negligible security regime unclonability does imply non-adaptive unbounded leakage-resilience, it does not imply LOCC leakage-resilience (as we suspected in Challenge 1): we show relative to a classical oracle that there exist schemes that are unclonable but do not satisfy even 1-round LOCC leakage-resilience. Similarly, due to Challenge 2, we are not able to show that unclonability implies even non-adaptive unbounded leakage-resilience in the constant security regime (such as CPA-style security). Thus, we need new techniques to show leakage-resilience.

2.1 LOCC Leakage-Resilience Property for Coset States

We first start by discussing coset states [16,27], which form the basis of our leakage-resilient schemes. We show a new LOCC leakage-resilience property for such states and use it to prove that various existing unclonable secure primitives [16,20] based on coset states are also LOCC leakage-resilient.

A *coset state* is the state defined as $|A_{s,s'}\rangle = \sum_{a\in A}(-1)^{\langle a,s'\rangle}|a+s\rangle$ where $A \subseteq \mathbb{F}_2^\lambda$ is a subspace of dimension $\dim(A) = \lambda/2$ and $s, s' \in \mathbb{F}_2^\lambda$. Coset states, when A, s, s' as above are randomly sampled, satisfy an important security notion called *monogamy-of-entanglement* (MoE) [16,27].

Monogamy-of-Entanglement Property of Coset States. In a monogamy-of-entanglement game, the adversary is presented with a randomly sampled coset state $|A_{s,s'}\rangle$ and is required to output two (possibly entangled) adversaries. Then, these adversaries are given the description of A, and they are required to *simultaneously* output vectors v, w such that $v \in A + s$ and $w \in A^\perp + s'$. Vidick and Zhang [27] show that no (unbounded) adversary can win the game above except with subexponentially small probability. Further, [16] also show a "computational" MoE property: based on computational assumptions, the winning probability of any (polynomial time) adversary in the game above is still negligible even when it is presented at the beginning of the game with an obfuscated program that allows it to query for membership in $A+s$ and $A^\perp+s'$. A variation implicitly used in [16,20], which we formally prove secure in our paper, presents the initial adversary with a *tuple* of coset states $|A_{i,s_i,s'_i}\rangle_{i\in[c(\lambda)]}$ (along with the

membership checking programs) and requires the created two adversaries to output vectors in either $A_i + s_i$ or $A_i^{\perp} + s'_i$ depending on the bits of the random challenge strings r_1 and r_2 presented to them.

LOCC Leakage-Resilience Property for Coset States. Our first technical contribution is to prove a new LOCC leakage-resilience property for coset states. Using this result, we also prove that the coset state based copy-protection schemes of [16,20] for public-key encryption, pseudorandom functions, and digital signatures also satisfy LOCC leakage-resilience. Note that this is non-trivial: as discussed above, unclonability does not imply LOCC leakage-resilience – there exists a PKE scheme that satisfies copy-protection that is provably not even 1-round LOCC leakage-resilient.

We define a leakage-resilience game for coset states as follows. Consider an adversary $\mathcal{A}$. For any (unbounded) number of rounds, $\mathcal{A}$ adaptively specifies measurements and obtains the measurement results on the secret state, starting with the state initialized to the coset state tuple $|A_{i,s_i,s'_i}\rangle_{i\in[c(\lambda)]}$. After the leakage phase is over, $\mathcal{A}$ is also given the descriptions of the subspaces A_i. Then, $\mathcal{A}$ is presented with a random challenge string $r \leftarrow \{0,1\}^{c(\lambda)}$, and is required to output vectors in $A_i + s_i$ or $A_i^{\perp} + s'_i$ depending on the i-th challenge bit, $(r)_i$. We show that any unbounded LOCC adversary wins this game with subexponentially small probability.

Connections to the Monogamy-of-Entanglement Property. While the LOCC leakage-resilience property might seem to be implied by the monogamy-of-entanglement property in a straightforward manner, in reality this does not immediately follow. Consider the following natural proposal for a reduction: in the MoE game, the initial adversary directly simulates the leakage adversary $\mathcal{A}$. However, observe that in the MoE game we need to output two adversaries that will need to answer the vector outputting challenges, while in LOCC leakage-resilience there is only one such adversary. Therefore, we would need to create two copies of the final internal quantum state of $\mathcal{A}$ to succeed in the MoE game. This is not straightforward, as discussed above (Challenge 1). However, our proof precisely manages to do this. Observe that during each round $\mathcal{A}$ takes as input its previous internal state and the latest leakage measurement outcome, and it produces a state for the next round and a choice of measurement circuit E_i. If we have sufficiently many (i.e., exponentially many) copies of its previous state, then by repeatedly running $\mathcal{A}$ on these copies we can obtain another copy of its next state conditioned on it producing the exact same choice of circuit E_i. Note that we want to obtain a copy conditioned on choosing the same leakage circuit E_i, since we start with a single copy of the coset state tuple and so we can measure it only once each round. We show that the multi-copy internal state generation procedure described above succeeds in some finite amount of time, and so, since we are working with unbounded adversaries, the reduction succeeds. See Sect. 4 for further details.

Moving to the Computational Setting. While the above result is a step forward, as we will later discuss all coset state-based constructions of [16,20] crucially rely on the obfuscated membership checking programs for $A_i + s_i$ and $A_i^{\perp} + s'_i$ for their correctness, and hence they also rely on the computational MoE property for their security. Therefore, analogous to MoE, we define a computational LOCC leakage-resilience game where the now computationally-bounded adversary $\mathcal{A}$ also receives at the beginning of the game the obfuscated programs that allow it to query for membership in $A_i + s_i$ and $A_i^{\perp} + s'_i$. Note that our reduction above from LOCC leakage-resilience to MoE might take exponentially long in general. Therefore, we are not able to utilize the same idea here to reduce the computational LOCC leakage-resilience to computational MoE. However, we show that the reduction of [16] from computational MoE to (information-theoretic) MoE that utilizes *subspace hiding* obfuscation shO [29] generalizes to the leakage-resilience setting. The argument mainly relies on using subspace hiding obfuscation to implement the membership checking programs, which can then be replaced with such programs for random superspaces of A_i by the security guarantee of shO. This eventually allows us to remove the membership checking programs, and hence the security reduces to the information-theoretic LOCC leakage-resilience game. Therefore, we are able to obtain a computational LOCC leakage-resilience property for coset states.

2.2 LOCC Leakage-Resilient PKE Using Coset States

We introduce the model of LOCC leakage-resilience for public-key encryption. The adversary adaptively chooses leakage/measurement circuits E_i for multiple rounds, and they are applied to the state of the key from the previous round (starting with the honestly generated secret key). After $\mathcal{A}$ finishes the leakage phase, it is presented with a challenge ciphertext that it needs to decrypt. Here, we can define two variations: random challenge message and CPA-style security. In the former, the challenge ciphertext is the encryption of a randomly sampled message, and the adversary needs to output the full message to win the game. We require that any adversary wins with at most negligible probability. In the latter, the adversary outputs two messages m_0, m_1 and the challenge ciphertext is the encryption of m_b where $b \leftarrow \{0,1\}$. The adversary wins if it can correctly guess the bit b, and we require that any adversary wins with probability at most $1/2 + \mathsf{negl}(\lambda)$, where λ is the security parameter.

Construction. Now, we move onto our construction. We show (via a new proof) that the coset state-based construction of [16] of a public-key encryption scheme with copy-protection also satisfies LOCC leakage-resilience. Let us first informally discuss this construction. We sample a tuple of coset states $|A_{i,s_i,s'_i}\rangle_{i\in[c]}$, and let this be the protected secret key. We also sample obfuscated programs that allow one to query if $v \in A_i + s_i$ and $v \in A_i^{\perp} + s'_i$, as in the computational LOCC leakage-resilience game. We set the public-key to be the tuple of these

programs. To encrypt a message m, we sample a random string r and let the ciphertext be $(i\mathcal{O}(P), r)$ where $i\mathcal{O}$ is an indistinguishability obfuscator and P is the following program: It takes as input vectors $(v_i)_{i\in[c]}$, and it checks if these vectors are in $A_i + s_i$ or $A_i^{\perp} + s'_i$ according to the bit $(r)_i$. If all vectors are in the correct cosets, then P outputs m; otherwise, P outputs $\perp$. It is easy to see that we can indeed construct this program using the public-key defined above.

Proving Security. For the intuition behind the security, first imagine we used an (ideal) classical oracle instead of $i\mathcal{O}$. If an LOCC leakage adversary $\mathcal{A}$ is able to correctly decrypt the challenge ciphertext, then $\mathcal{A}$ must be querying the oracle P at a tuple of vectors that pass the checks corresponding to r as described above. Hence, using this LOCC adversary and its queries, we can obtain vectors that are in correct cosets with respect to r. Observe that the checks above correspond exactly to the winning condition of the computational LOCC leakage-resilience game for coset states which we proved secure. Therefore, we can win that game, which is a contradiction. We conclude that the adversary cannot decrypt the challenge ciphertext.

Now we go back to the actual scheme, where $i\mathcal{O}$ is only an indistinguishability obfuscator. In this case, we first replace the ciphertext program with another one that computes the *canonical* versions of the input vectors and compares them to the canonical vectors of the cosets $A_i + s_i$ or $A_i^{\perp} + s'_i$ according to $(r)_i$. Since this is a compute-and-compare (CC) program, we can further replace it with its (distributionally) *virtual black-box* obfuscated version[3] Finally, by using the CC obfuscation security guarantee, we are able to extract vectors in the correct cosets if the adversary is successfully decrypting the ciphertext. This allows us to win the computational LOCC leakage-resilience game for the coset states, a contradiction. We have thus established the LOCC leakage-resilience of the public-key encryption scheme.

2.3 Leakage-Resilient Quantum Banks and Chosen-Message Leakage-Resilient Signatures

We also obtain leakage-resilience results for quantum money and chosen-message leakage-resilience security for signing keys. Let us first describe the setting for the former. We consider a quantum money scheme where the secret key of the bank is stored as a quantum state. An adversary obtains k banknotes and unbounded leakage on the secret key of the bank. Then, it is required to produce $k + 1$ banknotes.

We also define chosen-message leakage-resilience security for a signature scheme with quantum[4] signatures. In this case, the adversary obtains unbounded leakage on the signing key, and it outputs a forged signature for a message of its choice. We require that any efficient adversary succeed with at most negligible

[3] Such obfuscation for compute-and-compare can be constructed from LWE [28].
[4] As discussed, this notion is not possible with classical signatures.

probability. Our signature scheme and quantum bank scheme will be almost the same: In our signature scheme, the signature on a message m will be a quantum banknote and a classical signature on m. Then, it is easy to see that leakage-resilience for the signature scheme follows similarly to leakage-resilience for our quantum bank scheme. Therefore, in this section, we only discuss the latter.

Construction. Now we discuss our unbounded leakage-resilient quantum bank scheme, obtained in a black-box way from a *quantum lightning* scheme [29], which has a *candidate* construction relative to a classical oracle [7]. We set our quantum secret key to be the subspace state[5] $|A\rangle = \sum_{v \in A} |v\rangle$ where $A \subseteq \mathbb{F}_2^\lambda$ is a random subspace of dimension $\lambda/2$. We also create two oracles $\mathcal{O}_1, \mathcal{O}_2$ such that they accept a vector v and a quantum lightning bolt serial number sn, and they verify $v \in A$ or $v \in A^\perp$ respectively and output $H_i(sn)$ if v is correct, where H_1, H_2 are random functions. Here, we can think of $H_i(sn)$ as a classical signature on sn. To generate a banknote, we create a quantum lightning bolt, and then query the oracle $\mathcal{O}_1$ coherently on $|A\rangle$ and y to obtain $H_1(y)$. We then rewind and similarly obtain $H_2(y)$ from $\mathcal{O}_2$. A full banknote is a lightning bolt with some serial number y and the values $H_1(y), H_2(y)$. To verify a banknote, we simply verify the lightning bolt and the values $H_1(y), H_2(y)$ using a verification oracle for H_1, H_2.

Challenges. One might think that it is possible to obtain a chosen message secure leakage-resilient signature scheme (and thus a leakage-resilient quantum bank) in a black-box way using a random-challenge secure scheme and a provable randomness construction, such as a quantum lightning bolt with exponential security. However, such a scheme cannot be proven secure in a black-box way. In the reduction to the random-challenge security, the adversary for the chosen-message game will expect a serial number (which ensures the randomness) and a lightning bolt, whereas we will only receive a random message from our challenger, and it is not possible to create a lightning bolt with a particular serial number.

Proving Security. We briefly sketch our proof of unbounded leakage-resilience. We first claim that any adversary $\mathcal{A}$ that wins with non-negligible probability has non-negligible query weight on some $v \in A$ for the oracle $\mathcal{O}_1$. Suppose otherwise. Then, when $\mathcal{O}_1$ is replaced with an empty oracle, $\mathcal{A}$ still succeeds by the query weight lemma [12]. However, we claim that this is a contradiction. We leak once on the subspace state to obtain a classical leakage L, and we repeatedly run $\mathcal{A}$ (with empty oracle instead of $\mathcal{O}_1$) to obtain many valid banknotes, in

[5] Similarly to coset states, subspace states satisfy a property called *direct product hardness* [11]: no efficient adversary, even given oracle access to the membership checking program for A, can output $v \in A, w \in A^\perp$ given a single copy of $|A\rangle$.

particular valid signatures with respect to H_1. By quantum lightning security, all signatures will be on different serial numbers. Observe that we only used a fixed polynomial size *advice* L for H_1 and a verification oracle for it, but we produced any number of input-output pairs for the random function H_1, a contradiction. Thus, $\mathcal{A}$ indeed has non-negligible query weight on some $v \in A$ for the oracle $\mathcal{O}_1$, and similarly for $w \in A^\perp$ and the oracle $\mathcal{O}_2$. Thus, by leaking once on the subspace state, simulating $\mathcal{A}$ twice on the same leakage L and measuring a random query in both simulations, we obtain $v \in A, w \in A^\perp$, a contradiction.

2.4 The Relationship Between Leakage-Resilience and Unclonability

Our discussion will be based on public-key encryption, but other primitives also use similar ideas.

Random Challenge Unclonability $\implies$ *Unbounded Leakage-Resilience.* The naive reduction idea from leakage-resilience to unclonability, as discussed above, is as follows: to clone a key, we first leak on it and clone the leakage (which is classical). This approach indeed works in the random-challenge unbounded leakage-resilience setting. However, it does not work in the constant security regime, for example in CPA-style security: the probability of two freeloaders succeeding will be $(1/2 + 1/p)^2 \approx 1/4$, whereas the baseline is $1/2$ in CPA-style security.

Unclonability $\not\implies$ 1*-Round LOCC Leakage-Resilience.* We only showed that random challenge unclonability implies unbounded leakage-resilience, rather than LOCC leakage-resilience. One might wonder whether this is a weakness of our proof techniques. We show that this is inherent: we prove that there exists a scheme that satisfies both random-challenge and CPA-style unclonability, but does not satisfy even 1-round LOCC leakage-resilience. We now describe this scheme, which is based[6] on *equivocal* collision-resistant hash functions, which has a candidate construction relative to a classical oracle [7]. In the scheme, a ciphertext will be of the form $(r, F_2(r) \oplus F_3(r) \oplus m)$. We define the quantum secret key to be a tuple of subspace states $|A_i\rangle_{i\in[\ell]}$. We also define the oracles $\mathcal{O}_1, \mathcal{O}_2$. The oracle $\mathcal{O}_1$ takes as input x, y, r and vectors $(v_i)_{i\in[\ell]}$. It first verifies that x is a preimage (with respect to the hash function H) of y and starts with 0, and then verifies the vectors with respect to the subspaces A_i and the *basis* y. If all checks pass, it outputs $F_2(r)$. The oracle $\mathcal{O}_2$ is similarly defined, but it checks if the first bit of x is 1, and it outputs $F_3(r)$ at the end. Observe that, intuitively, an adversary will need to query both oracles with *correct* inputs to be able to decrypt a ciphertext.

A 1-round leakage adversary simply creates a superposition $\sum_{x:H(x)=y}|x\rangle$, and chooses the leakage circuit to be measurements in the computational or Hadamard basis (according to the i-th bit of y). Thus, the leakage gives the adversary $(v_i^*)_{i\in[\ell]}$ such that $v_i^* \in A_i$ if $(y)_i = 0$ and $v_i^* \in A_i^\perp$ if $(y)_i = 1$. The adversary can then decrypt any ciphertext by equivocating (coherently) the

[6] We give a concrete construction based on equivocal CRHFs. Using similar ideas, any quantum lightning scheme based on classical oracles suffices.

superposition state to first 0 and querying $\mathcal{O}_1$, then rewinding and repeating for 1 and querying $\mathcal{O}_2$. It is easy to see using the correctness of the vectors v_i^* that the adversary obtains $F_2(r), F_3(r)$ with overwhelming probability and can successfully decrypt.

However, we argue that the scheme is unclonable. We first note that the main difference stems from the fact that in the LOCC leakage-resilience game it was sufficient to construct a single successful decryptor, whereas here we need to construct two to break security. As argued above, a freeloader will need to query $\mathcal{O}_1$ and $\mathcal{O}_2$ at correct $x, y, (v_i)_{i\in[\ell]}$. With overwhelming probability, the two freeloaders will query $\mathcal{O}_1, \mathcal{O}_2$ respectively at different y, since otherwise we would have a collision[7]. Thus, we extract $(v_i)_{i\in\ell}$ and $(w_i)_{i\in\ell}$ from the two freeloaders, and since they are correct with respect to $y_1 \neq y_2$, there is $i \in [\ell]$ such that $v_i \in A_i$ and $w_i \in A_i^{\perp}$ (or vice versa), a contradiction to *direct product hardness* of the subspace state A_i. In the formal proof, to deal with extracting simultaneously from entangled freeloaders, we use projective implementations [3].

LOCC Leakage-Resilience $\not\Longrightarrow$ Unclonability. We obtain this result by showing that a modified version of our LOCC leakage-resilient PKE scheme is still LOCC leakage-resilient. In this case, the secret key of the scheme consists of k independently sampled coset state tuples, and a ciphertext program accepts any of them as valid. We show that this scheme is still LOCC leakage-resilient, while it is trivially clonable: the cloning adversary gives one coset state tuple to one freeloader and another one to the other freeloader.

2.5 Intrusion-Detection

We discuss how to construct an intrusion-detection scheme from any publicly verifiable key leasing (i.e., certified key deletion) scheme for a primitive. As an example, we will elaborate on public-key encryption.

We discuss how to construct public-key encryption schemes that support intrusion-detection for arbitrary attacks on the decryption key $\mathsf{R}_{\mathsf{dec}}$. More precisely, the PKE scheme generates a public key pk, a test key tk (used to test whether an intrusion occurred), and a (quantum) decryption key $\mathsf{R}_{\mathsf{dec}}$. An adversary is given $(pk, tk, \mathsf{R}_{\mathsf{dec}})$, and it can arbitrarily act on the quantum part to produce a quantum state $\mathsf{R}_{\mathsf{adv}}$ and two challenge messages m_0 and m_1. Note that this may change the state in register $\mathsf{R}_{\mathsf{dec}}$. Before the distinguishing game proceeds, a intrusion-detection step is run and the adversary automatically loses if its presence is detected, i.e., if $\mathsf{TestIntrusion}(tk, \mathsf{R}_{\mathsf{dec}}) = \mathsf{INTRUSION}$. If no intrusion is detected, we want to guarantee that it is not possible to distinguish between $\mathsf{Enc}(pk, m_0)$ and $\mathsf{Enc}(pk, m_1)$ given $(\mathsf{R}_{\mathsf{leak}}, m_0, m_1, tk, pk)$ with probability negligibly close to the baseline $\frac{1}{2}\Pr[\mathsf{TestIntrusion}(tk, \mathsf{R}_{\mathsf{dec}}) = \mathsf{NO\ INTRUSION}]$.

[7] Note that inputs x to $\mathcal{O}_1$ are required to start with 0, and inputs x to $\mathcal{O}_2$ are required to start with 1. Thus, if we had the same y and valid x_1, x_2 for both oracles, we would have $H(x_1) = H(x_2)$ and $(x_1)_1 = 0, (x_2)_1 = 1$ (thus $x_1 \neq x_2$).

We construct PKE schemes with these guarantees by establishing a connection to secure key leasing [8–10,18]. We start with the notion of a PKE scheme with secure key leasing, which features an additional *deletion procedure* that, given the secret decryption key $\mathsf{R}_{\mathsf{dec}}$, produces a certificate *cert* which should certify that this key was indeed deleted. Roughly speaking, this scheme satisfies the property that an adversary which is able to produce a valid certificate *cert* based on $\mathsf{R}_{\mathsf{dec}}$ (validity of *cert* is checked by a Verify procedure using a certificate validation key *cvk*) cannot distinguish between the ciphertexts $\mathsf{Enc}(pk, m_0)$ and $\mathsf{Enc}(pk, m_1)$ using the leftover state. PKE schemes with secure key leasing have been recently constructed from any post-quantum PKE scheme [5][8] or post-quantum indistinguishability obfuscation [9].

We show that we can construct a PKE scheme that supports intrusion-detection from a PKE scheme with secure key leasing. Starting with a PKE scheme with secure key leasing, we construct a $\mathsf{TestIntrusion}$ procedure which essentially tries to produce a deletion certificate for the secret decryption key $\mathsf{R}_{\mathsf{dec}}$, and outputs $\mathsf{NO\ INTRUSION}$ if it succeeds. Intuitively, we can argue intrusion-detection security as follows: If an adversary has obtained information that allows it to distinguish ciphertexts, then we should fail to produce a valid deletion certificate using our leftover state. Otherwise, one can create a lessee attacker against the key leasing security that simulates the intrusion adversary on their key, produces a valid deletion certificate using the leftover state, and still succeeds in distinguishing ciphertexts using the stolen state. However, the major problem with this approach is reusability: even when there is no attack, we destroy our key when we test for leakage, since we produce a deletion certificate.

Crucially, note that producing a valid deletion certificate using an undisturbed key succeeds with overwhelming probability. Therefore, using the gentle measurement lemma [1], we are able to construct an algorithm for producing a deletion certificate in such a way that we can rewind our algorithm afterwards.

Using similar techniques, we can also build digital signatures, PRFs, functional encryption, and indistinguishability obfuscation schemes supporting intrusion-detection. More generally, we show that the notion of intrusion-detection is *equivalent* to key leasing/certified deletion.

3 Preliminaries

We write *QPT* to mean *quantum polynomial time*. We refer the reader to [21] for various cryptographic preliminaries, and to [15,24] for preliminaries on indistinguishability obfuscation $i\mathcal{O}$ and puncturable pseudorandom functions (PRF).

3.1 Quantum Lightning

Definition 1 (Quantum Lightning [29]). *A quantum lightning scheme consists of the following algorithms.*

[8] This scheme unfortunanely lacks public verifiability, which is crucial for intrusion-detection since the adversary gets the complete state of the honest party, including the verification key.

- $\mathsf{Bolt}(1^\lambda)$: *Samples a lightning bolt with a serial number.*
- $\mathsf{Ver}(sn, R)$: *Takes in a supposed bolt register R and a serial number sn. Outputs 1 if R is a valid bolt with serial number sn. Otherwise, outputs 0.*

Correctness. We require that any honestly generated bolt passes the verification with overwhelming probability.

Security. We require that for any QPT adversary $\mathcal{B}$, when we run $(sn, R_1, R_2) \leftarrow \mathcal{B}$, $b \leftarrow \mathsf{Ver}(sn, R_1)$ and $b' \leftarrow \mathsf{Ver}(sn, R_2)$, we have $\Pr[b = b' = 1] \leq \mathsf{negl}(\lambda)$.

3.2 Equivocal Collision-Resistant Hash Functions

Definition 2 (Equivocal Collision-Resistant Hash Functions (CRHF) [7]). *An equivocal collision-resistant hash function consists of the following algorithms:*

- $\mathsf{Gen}(crs)$: *Takes as input a common reference string and outputs a hash value y, a predicate p and a quantum inversion key $\mathsf{R}_{\mathsf{key}}$.*
- $\mathsf{Equiv}(\mathsf{R}_{\mathsf{key}}, b)$: *Takes as input a quantum inversion key $\mathsf{R}_{\mathsf{key}}$ and a bit b, and outputs a pre-image x.*

We require collision-resistance (in the usual way) and correctness for Equiv*: for $b \in \{0,1\}$, $\Pr[H_{crs}(x) = y \wedge p(x) = b : \begin{smallmatrix} y, p, \mathsf{R}_{\mathsf{key}} \leftarrow \mathsf{Gen}(crs) \\ x \leftarrow \mathsf{Equiv}(\mathsf{R}_{\mathsf{key}}, b) \end{smallmatrix}]$.*

Theorem 11 ([7]). *There exists a* candidate *equivocal CRHF relative to a classical oracle. More formally, there exists a dimension $\lambda/2$ affine ordered partitioning $P = (A_y)_{y \in \{0,1\}^{\lambda/2}}$ of the space $\mathbb{F}_2^\lambda$ such that the hash function $H_P : \mathbb{F}_2^\lambda \to \{0,1\}^{\lambda/2}$ that maps a vector $v \in \mathbb{F}_2^\lambda$ to the affine subspace (with respect to P) it is in, is an equivocal CRHF candidate relative to the classical oracles $\mathcal{O}_P, \mathcal{O}_P^\perp$ defined as:*

1. $\mathcal{O}_P(x) = H_P(x)$,
2. $\mathcal{O}_P^\perp(x, y)$: *Let A be the subspace associated with the affine space $A_y \in P$. Output 1 if and only if $x \in A^\perp$.*

4 Coset States

In this section, we start by giving the definition of *coset states* [16,27] that we utilize in our constructions and state the monogamy-of-entanglement property they satisfy. Then, we prove an LOCC leakage-resilience theorem for coset states.

Definition 3 ([16]). *For a subspace $A \subseteq \mathbb{F}_2^n$ and vectors $s, s' \in \mathbb{F}_2^n$, we define $|A_{s,s'}\rangle$, the* coset state *associated with A, s, s', to be $|A_{s,s'}\rangle = \sum_{a \in A} \frac{1}{\sqrt{|A|}} (-1)^{\langle s', a \rangle} |a + s\rangle$.*

We usually write $A + s$ to denote both the coset $A + s$ and the program that takes as input a vector $v \in \mathbb{F}_2^n$ and outputs 1 if and only if $v \in A + s$. The distinction will be clear from the context. We will write $\mathsf{Can}_A(v)$ to denote the lexicographically smallest element in the coset $A + v$, and call it the canonical element. Now we state the monogamy-of-entanglement (*MoE*) properties coset states satisfy.

Theorem 12 (Strong Monogamy-of-Entanglement Property for Coset States [16]). *Consider the following game between an adversary tuple $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$ and the challenger.*

$\underline{\mathsf{MoE}(\lambda, \mathcal{A})}$

1. *Sample uniformly at random a subspace A of $\mathbb{F}_2^\lambda$ of dimension $\frac{\lambda}{2}$ and two elements $s, s' \leftarrow \mathbb{F}_2^\lambda$.*
2. *Submit $|A_{s,s'}\rangle$ to $\mathcal{A}_0$.*
3. *$\mathcal{A}$ outputs two (possibly entangled) registers R_1, R_2.*
4. *For $\ell \in \{1, 2\}$, run $v_\ell \leftarrow \mathcal{A}_\ell(R_\ell, A)$.*
5. *Output 1 if and only if $v_1 \in A + s$ and $v_2 \in A^\perp + s'$.*

Then, there exists a constant $C_{\mathsf{MoE}} > 0$ such that for any adversary tuple $(\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$, $\Pr[\mathsf{MoE}(\lambda, \mathcal{A}) = 1] \leq 2^{-\lambda^{C_{\mathsf{MoE}}}}$ for all sufficiently large λ.

We show the security of a similar game which was used implicitly in previous work [16,20]. Let $\mathsf{CosetGen}(1^\lambda)$ be the algorithm that samples $c(\lambda) = 3 \cdot \lambda^{\lceil 1/C_{\mathsf{MoE}} \rceil}$ independent cosets $(A_i, s_i, s'_i)_{i \in [c(\lambda)]}$ of $\mathbb{F}_2^\lambda$ of dimensions $\lambda/2$, and call its output a coset tuple.

Theorem 13 (Strong Monogamy-of-Entanglement Property for Coset States - Multiple Challange Version). *Consider the following game between an adversary tuple $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$ and the challenger.*

$\underline{\mathsf{MoE} - \mathsf{MultiChal}(\lambda, \mathcal{A})}$

1. *Sample $(A_i, s_i, s'_i)_{i \in [c(\lambda)]} \leftarrow \mathsf{CosetGen}(1^\lambda)$.*
2. *Submit $\left\{|A_{i,s_i,s'_i}\rangle\right\}_{i \in [c(\lambda)]}$ to $\mathcal{A}_0$.*
3. *$\mathcal{A}$ outputs two (possibly entangled) registers R_1, R_2.*
4. *Sample $r_1 \leftarrow \{0,1\}^{c(\lambda)}$ and $r_2 \leftarrow \{0,1\}^{c(\lambda)}$.*
5. *For $\ell \in \{1, 2\}$, run $(v_{\ell,i})_{i \in [c(\lambda)]} \leftarrow \mathcal{A}_\ell(R_\ell, r_\ell, (A_i)_{i \in [c(\lambda)]})$.*
6. *For $\ell \in \{1, 2\}$ and all $i \in [c(\lambda)]$, check if $v_{\ell,i} \in A_i + s_i$ if $(r_\ell)_i = 0$ and if $v_{\ell,i} \in A_i^\perp + s'_i$ if $(r_\ell)_i = 1$. Output 1 if and only if all the checks pass. Otherwise, output 0.*

Then, there exists a constant $C_{\mathsf{MoE-MultChal}} > 0$ such that for any adversary tuple $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2)$, $\Pr[\mathsf{MoE} - \mathsf{MultiChal}(\lambda, \mathcal{A}) = 1] \leq 2^{-\lambda^{C_{\mathsf{MoE-MultChal}}}}$ for all sufficiently large λ.

Proof. See the full version [15].

4.1 LOCC Leakage-Resilience

In this section, we show an *LOCC leakage-resilience* property for coset states. We start by defining an LOCC leakage adversary.

Definition 4 (LOCC Leakage Adversary). *An LOCC leakage adversary is a* stateful *quantum algorithm $\mathcal{A}$ that (adaptively) specifies*[9] *quantum leakage circuits for multiple rounds to leak on a secret quantum state. More formally, we will consider the following experiment between $\mathcal{A}$ and a challenger, for a distribution of quantum secret states and public parameters (denoted pp) induced by a quantum algorithm* Setup.

$\underline{\mathsf{LEAKAGE} - \mathsf{EXP}(1^\lambda)}$

1. *The challenger executes $\mathsf{R}_0, pp \leftarrow \mathsf{Setup}(1^\lambda)$ and submits pp to $\mathcal{A}$.*
2. *For multiple rounds, $\mathcal{A}$ specifies a quantum leakage circuit E_i that takes input a quantum register and outputs a classical string and an updated quantum register. For each round, the challenger executes $L, \mathsf{R}_i \leftarrow E_i(\mathsf{R}_{i-1})$ and submits the classical string L to $\mathcal{A}$.*
3. *At the end of the leakage phase, the final state of the adversary $\mathcal{A}$ is output.*

We will require that the leakage circuits specified by $\mathcal{A}$ are consistent in the sense that the input size of E_i is the same as the size of the quantum register R_{i-1}. In general, the leakage adversary $\mathcal{A}$ will continue to complete a challenge in a cryptographic game with the state it has created and output at the end of the leakage experiment above. Note that there is no bound[10] *on the number of rounds leakage for an* LOCC leakage adversary. *If an adversary is allowed to leak for only k-rounds, we call it a* k-round LOCC leakage adversary.

We define an unbounded classical leakage adversary[11] *to be a pair of quantum algorithms $(E_0, \mathcal{A})$ where $\mathcal{A}$ only obtains the single shot classical leakage L where $L, \mathsf{R}' \leftarrow E_0(\mathsf{R}_0, pp)$.*

Remark 1. It is easy to see that LOCC leakage-resilience is the strongest setting, while (non-adaptive) unbounded classical leakage is the weakest, and for any k, we have that $(k+1)$-round LOCC leakage-resilience is stronger than k-round LOCC leakage-resilience.

An equivalent way of defining LOCC leakage adversaries is as a pair of adversaries that execute an LOCC (local operations and classical communication protocol) - see the full version [15].

[9] We assume that the adversary outputs the classical description of an appropriate quantum circuit in a canonical representation.

[10] In the computational setting, this will implicitly be any (not a-priori bounded) polynomial.

[11] We also call this *non-adaptive unbounded classical leakage adversary*, since the leakage circuit is not specified by $\mathcal{A}$ after getting the public parameters. However, we note that this is still somewhat adaptive since the leakage circuit E_0 does get the public parameters.

Now we move on the leakage-resilience property for coset states. In the LOCC leakage game for coset states, the leakage adversary will obtain LOCC leakage on a coset state tuple. After the leakage is completed, the adversary is presented with a random challenge string r, and is required to produce vectors in correct cosets depending on r.

Theorem 14 (LOCC Leakage Property for Coset States). *Consider the following game between an LOCC leakage adversary $\mathcal{A}$ (Definition 4) and the challenger.*

$\underline{\mathsf{Coset}-\mathsf{LOCC}(\lambda,\mathcal{A})}$

1. *Sample* $(A_i, s_i, s'_i)_{i\in[c(\lambda)]} \leftarrow \mathsf{CosetGen}(1^\lambda)$.
2. *$\mathcal{A}$ obtains leakage on* $\left\{|A_{i,s_i,s'_i}\rangle\right\}_{i\in[c(\lambda)]}$.
3. *The challenger samples* $r \leftarrow \{0,1\}^{c(\lambda)}$ *and submits* $(A_i)_{i\in[c(\lambda)]}$ *and* r *to* $\mathcal{A}$.
4. *$\mathcal{A}$ outputs* $(v_i)_{i\in[c(\lambda)]}$.
5. *For all* $i \in [c(\lambda)]$, *check if* $v_i \in A_i + s_i$ *if* $(r)_i = 0$ *and if* $v_i \in A_i^\perp + s'_i$ *if* $(r)_i = 1$. *Output* 1 *if and only if all the checks pass. Otherwise, output* 0.

Then, there exists a constant $C_{\mathsf{LOCC}} > 0$ *such that for any LOCC leakage adversary* $\mathcal{A}$, $\Pr[\mathsf{Coset}-\mathsf{LOCC}(\lambda,\mathcal{A})=1] \leq 2^{-\lambda^{C_{\mathsf{LOCC}}}}$ *for all sufficiently large* λ.

We will prove the above result through a reduction to $\mathsf{MoE}-\mathsf{MultiChal}$, which we briefly sketch. In the reduction, since the adversary for $\mathsf{MoE}-\mathsf{MultiChal}$ will be in possession of the coset state tuple, it can simulate the leakage adversary $\mathcal{A}$ to obtain its final state. However, the MoE adversary needs to produce two registers that are capable of answering the challenges correctly simultaneously. Hence, during the simulation of the LOCC leakage, we run $\mathcal{A}$ many times each round to produce multiple copies of its state for the next round, culminating in two copies of its final state, which then we output. While the probability of obtaining another copy of the adversary's state during a round might be arbitrarily small, we show that on average this is not the case.

Proof. Suppose for a contradiction that there exists a $g(n)$-round LOCC leakage adversary $\mathcal{A}$ that wins $\mathsf{Coset}-\mathsf{LOCC}$ with probability $2^{-0.1\lambda^{C_{\mathsf{MoE}-\mathsf{MultChal}}}}$. Without loss of generality, assume that the size of the leakage circuits output by the adversary each round is the same length and denote it as $k(n)$. We will construct an adversary $\mathcal{A}' = (\mathcal{A}'_0, \mathcal{A}'_1, \mathcal{A}'_2)$ that wins $\mathsf{MoE}-\mathsf{MultiChal}$ with probability $2^{-0.3\lambda^{C_{\mathsf{MoE}-\mathsf{MultChal}}}}$.

Let $\mathcal{P}$ denote the random variable that contains the transcript (i.e., the leakage circuit description and the classical leakage output by it) of the LOCC leakage experiment played by $\mathcal{A}$ during $\mathsf{Coset}-\mathsf{LOCC}$ and let $\mathcal{P}^{-1}$ denote the same but with the last leakage string removed. For some fixed value $(\ell^{-1}, E_{g(n)})$ of $\mathcal{P}^{-1}$, let $\rho_{\ell^{-1},E_{g(n)}}$ denote the final state of $\mathcal{A}$, i.e., its state right before it receives the last leakage string $\ell_{g(n)}$ and the challenge string r, conditioned on $\mathcal{P}^{-1} = (\ell^-1, E_{g(n)})$. Note that the final state of $\mathcal{A}$ does not depend on $\ell_{g(n)}$. Define deterministic $f(w)$ as $f(w) = (A_i)_{i\in[c(\lambda)]}$ where $(A_i, s_i, s'_i)_{i\in[c(\lambda)]} = \mathsf{CosetGen}(1^\lambda; w)$

and similarly deterministic predicate P as the function that outputs 1 if and only if the given vectors are in the correct cosets according to r. Then, the winning probability of $\mathcal{A}$ can be written as

$$\mathbb{E}_{(w,(\ell,E_{g(n)}))\leftarrow(W,\mathcal{P})}\left[\Pr_{r\leftarrow\{0,1\}^{c(\lambda)}}[P(\mathcal{A}(\rho_{\ell^{-1},E_{g(n)}},\ell_{g(n)},f(w),r),w,r)=1]\right]. \quad (1)$$

If we had two copies of $\rho_{\ell^{-1},E_{g(n)}}$ for the same coset state tuple and transcript $\ell^{-1},E_{g(n)}$, and ran the last step of $\mathcal{A}$ twice independently on independent challenge strings r_1,r_2, we would have that the probability of both copies *winning* simultaneously is $\mathbb{E}_{(w,(\ell,m))\leftarrow(W,\mathcal{P})}\Big[\left(\Pr_{r_1\leftarrow\{0,1\}^{c(\lambda)}}[P(\mathcal{A}(\rho_{l^{-1},m},\ell_{g(n)},f(w),r_1),w,r_1)=1]\right)\cdot\left(\Pr_{r_2\leftarrow\{0,1\}^{c(\lambda)}}[P(\mathcal{A}(\rho_{l^{-1},m},\ell_{g(n)},f(w),r_2),w,r_2)=1]\right)\Big] = \mathbb{E}_{(w,(\ell,m))\leftarrow(W,\mathcal{P})}\left[\left(\Pr_{r\leftarrow\{0,1\}^{c(\lambda)}}[P(\mathcal{A}(\rho_{l^{-1},m},\ell_{g(n)},f(w),r),w,r)=1]\right)^2\right]$. Then, since we have (1) $> 2^{-0.1\lambda^{C_{\mathsf{MoE-MultChal}}}}$, we get that the expression above is $> 2^{-0.2\lambda^{C_{\mathsf{MoE-MultChal}}}}$ by Jensen's inequality.

We will construct an adversary $\mathcal{A}'_0$ for $\mathsf{MoE}-\mathsf{MultiChal}$ such that given only a single copy of the coset state tuple, it produces two copies of $\rho_{\ell^{-1},E_{g(n)}}$ with probability at least 1/2 (no matter which fixed value of the transcript we condition on). Note that we produce two copies of $\rho_{\ell^{-1},E_{g(n)}}$ for the same $\ell^{-1},E_{g(n)}$, since we only have one copy of the coset state tuple and executing a leakage round modifies this state. Therefore, we can run a leakage circuit once each round, hence we run it on a single choice of a leakage circuit E_i by $\mathcal{A}$ and receive only a single leakage ℓ_i each round. $\mathcal{A}'_0$ outputs $((\rho_{\ell^{-1},E_{g(n)}},\ell_{g(n)}),(\rho_{\ell^{-1},E_{g(n)}},\ell_{g(n)}))$ as its bipartite state output. Finally, $\mathcal{A}'_1$ and $\mathcal{A}'_2$ both simulate $\mathcal{A}$ on the state they receive, along with the subspace descriptions and the random challenge string they receive from the challenger. Then, winning corresponds exactly to Sect. 4.1, therefore we get that $\mathcal{A}'$ wins $\mathsf{MoE}-\mathsf{MultiChal}$ with probability $2^{-0.2\lambda^{C_{\mathsf{MoE-MultChal}}}}\cdot\frac{1}{2} > 2^{-\lambda^{C_{\mathsf{MoE-MultChal}}}}$, which is a contradiction by Theorem 13.

Now we will show how to construct such an adversary. First, note that the final state (before receiving the final leakage string) ρ of the main adversary is output together with a leakage circuit $E_{g(n)}$, by running $\mathcal{A}$ on its previous state $\rho_{g(n)-1}$ and the previous leakage string $\ell_{g(n)-1}$. In turn, $\rho_{g(n)-1}$, $E_{g(n)-1}$ and so on, all the way down to the initial state of $\mathcal{A}$ are sampled similarly. Consider any input σ to $\mathcal{A}$, i.e. a previous state and a leakage circuit description[12]. Let $\sum_{x\in\{0,1\}^{k(n)}} p_x|x\rangle\langle x|\otimes\tau_x$ denote the output of $\mathcal{A}(\sigma)$. Then, we claim that given $a(n)$ copies of σ and a fixed value x, we can produce $d(n)$ extra copies of τ_x for any $d(n)$ with probability $(1/2)^{1/g(n)}$ averaged over x, where $a(n)=\frac{2^{k(n)+1}\cdot d^2(n)}{1-(1/2)^{(1/g(n))-1}}$. Starting with $d(n)=1$ for the last round, we can calculate the number of copies needed all the way down to the first level. While grows large with every round, the total number of copies of the initial state needed is bounded since we have

[12] We bundle the leakage circuit description in the state σ.

$g(n)$ rounds. Therefore, we can construct a valid $\mathcal{A}'_0$ as follows. For each round, it first simulates $\mathcal{A}$ to obtain a leakage circuit E and a state. Then, it keeps running $\mathcal{A}$ repeatedly until obtains the same leakage circuit E again, in which case it has also obtained the required copy of the state. It repeats this procedure many times to obtain sufficiently many copies for the next round. Finally, it runs E on the coset state tuple to produce the leakage. Repeating this simulation until the last round shows that we can obtain two copies of $\rho_{\ell^{-1},E_{g(n)}}$ in a bounded amount of time. Note that by above, the many copy preparation procedure succeeds with probability $(1/2)^{1/g(n)}$ for each round, independently of succeeding in the previous rounds since we made the claim above for any input σ. Hence, we will obtain two copies of $\rho_{\ell^{-1},E_{g(n)}}$ with probability $1/2$ as desired.

Lastly, we prove our claim that $a(n)$ copies of the input to $\mathcal{A}$ is sufficient to produce $d(n)$ extra copies of its output. While the desired output E that we want for it to reoccur might have arbitrarily small probability, in which case it would take arbitrarily long to obtain the same state again, this happens *rarely*. More formally, define the set GOOD to be all $E \in \{0,1\}^{k(n)}$ such that $p_E > 2^{-k(n)}(1-(1/2)^{(1/g(n))-1})$ Then, a simple calculation shows that $\sum_{E\in\mathsf{GOOD}} p_E > (1/2)^{(1/g(n))-1}$. We have that the probability of getting the first outcome E again $d(n)$ many times in $a(n)$ trials, averaged over all E, is at least $\sum_{E\in\mathsf{GOOD}} p_x(1-(1-p_x)^{a(n)/d(n)})^{(d(n))}$. A simple calculation shows that this value is at least $(1/2)^{1/g(n)}$ as desired.

Finally, we can also define a computational version of $\mathsf{Coset}-\mathsf{LOCC}$, denoted $\mathsf{Coset}-\mathsf{CompLOCC}$, where the adversary $\mathcal{A}$ receives obfuscated membership checking programs $i\mathcal{O}(A_i+s_i), i\mathcal{O}(A_i^{\perp}+s'_i)$ for the cosets, along with the coset states, at the beginning of the game.

Theorem 15. *If we assume the existence of subexponentially-secure $i\mathcal{O}$ and one-way functions, then there exists a constant $C_{\mathsf{CompLOCC}} > 0$ such that for any QPT LOCC leakage adversary $\mathcal{A}$ we have that*

$$\Pr[\mathsf{Coset}-\mathsf{CompLOCC}(\lambda,\mathcal{A})=1] \leq 2^{-\lambda^{C_{\mathsf{CompLOCC}}}}$$

for all sufficiently large λ.

We refer the reader to the full version [15] for the proof.

5 Public-Key Encryption with Key Protection

In this section, we introduce the concept of public-key encryption schemes with key protection and define various security models for it. A public-key encryption schemes with key protection is a PKE with quantum secret keys and classical ciphertexts and classical public keys, where we require correctness and CPA security as usual.

We now introduce the notion of LOCC leakage-resilience. Similar to anti-piracy, we can define two variants: random challenge message and CPA style.

Definition 5 (LOCC Leakage-Resilience for Public-Key Encryption). *Consider the following game between the challenger and an LOCC leakage adversary $\mathcal{A}$ (Definition 4).*

$\underline{\mathsf{PKE} - \mathsf{LOCC} - \mathsf{CPA}(\lambda, \mathcal{A})}$

1. *Sample* $pk, sk \leftarrow \mathsf{PKE.Setup}(1^\lambda)$.
2. *Sample* $\mathsf{R}_{\mathsf{key}} \leftarrow \mathsf{PKE.QKeyGen}(sk)$.
3. *Submit* pk *to* $\mathcal{A}$.
4. $\mathcal{A}$ *obtains leakage on* $\mathsf{R}_{\mathsf{key}}$.
5. $\mathcal{A}$ *outputs two messages* m_0, m_1.
6. *Challenger samples* $b' \leftarrow \{0,1\}$ *and* $ct \leftarrow \mathsf{PKE.Enc}(pk, m_b)$ *and submits* ct *to* $\mathcal{A}$.
7. $\mathcal{A}$ *outputs a guess* $b' \in \{0,1\}$.
8. *The challenger outputs* 1 *if and only if* $b' = b$.

A public-key encryption scheme PKE *with key protection is said to satisfy* CPA-style LOCC leakage-resilience *if for any QPT LOCC adversary* $\mathcal{A}$*,*

$$\Pr[\mathsf{PKE} - \mathsf{LOCC} - \mathsf{CPA}(\lambda, \mathcal{A}) = 1] \leq \frac{1}{2} + \mathsf{negl}(\lambda).$$

We also define $\mathsf{PKE} - \mathsf{LOCC} - \mathsf{Guess}$ *to be the same as* $\mathsf{PKE} - \mathsf{LOCC} - \mathsf{CPA}$ *except that the adversary is given* $ct \leftarrow \mathsf{PKE.Enc}(pk, m)$ *where* m *is a uniformly random message, and the adversary is required to output* m *in full.* PKE *is said to satisfy* random challenge message LOCC leakage-resilience *if for all QPT LOCC leakage adversaries* $\mathcal{A}$*,*

$$\Pr[\mathsf{PKE} - \mathsf{LOCC} - \mathsf{Guess}(\lambda, \mathcal{A}) = 1] \leq \frac{1}{|\mathcal{M}|} + \mathsf{negl}(\lambda).$$

5.1 Coset State-Based Construction

In this section, we show that the anti-piracy secure public-key encryption scheme of [16] based on coset states (Sect. 4) also satisfies CPA-style LOCC leakage-resilience. For completeness, we first recall the construction of [16], slightly modified to match our notation and the parameters we require for LOCC leakage-resilience.

Assume the existence of following schemes.

- $i\mathcal{O}$, subexponentially secure indistinguishability obfuscation scheme,
- CCObf, compute-and-compare obfuscation scheme[13] [28] for $2^{-\lambda^{0.5 \cdot c_{\mathsf{CompLOCC}}}}$-unpredictable distributions.
- Subexponentially-secure one-way functions.

$\underline{\mathsf{PKE.Setup}(1^\lambda)}$

[13] This is not needed for the construction but it is needed for the security proof.

1. Sample $(A_i, s_i, s'_i)_{i\in[c(\lambda)]} \leftarrow \mathsf{CosetGen}(1^\lambda)$.
2. Set $sk = (A_i, s_i, s'_i)_{i\in[c(\lambda)]}$.
3. For $i \in [c(\lambda)]$,
   (a) Sample $\mathsf{OP}^0_i \leftarrow i\mathcal{O}(A_i + s_i)$.
   (b) Sample $\mathsf{OP}^1_i \leftarrow i\mathcal{O}(A^\perp_i + s'_i)$.
4. Set $pk = (\mathsf{OP}^0_i, \mathsf{OP}^1_i)_{i\in[c(\lambda)]}$.
5. Output (pk, sk).

$\underline{\mathsf{PKE.QKeyGen}(sk)}$

1. Parse $(A_i, s_i, s'_i)_{i\in[c(\lambda)]} = sk$.
2. Output $\left(|A_{i,s_i,s'_i}\rangle\right)_{i\in[c(\lambda)]}$.

$\underline{\mathsf{PKE.Enc}(pk, m)}$

1. Parse $(\mathsf{OP}^0_i, \mathsf{OP}^1_i)_{i\in[c(\lambda)]} = pk$.
2. Sample $r \leftarrow \{0,1\}^{c(\lambda)}$.
3. Sample $\mathsf{OPCt} \leftarrow i\mathcal{O}(\mathsf{PCt})$, where PCt is the following program.

   $\underline{\mathsf{PCt}(u_1, \ldots, u_{c(\lambda)})}$
   Hardcoded: $(\mathsf{OP}^0_i, \mathsf{OP}^1_i)_{i\in[c(\lambda)]}, m, r$
   (a) For $i \in [c(\lambda)]$, check if $\mathsf{OP}^0_i(u_i) = 1$ if $(r)_i = 0$ and if $\mathsf{OP}^1_i(u_i) = 1$ if $(r)_i = 1$.
   (b) Output m if all the checks pass. Otherwise, output $\perp$.

4. Output (OPCt, r).

$\underline{\mathsf{PKE.Dec}(\mathsf{R}_{\mathsf{key}}, ct)}$

1. Parse $((R_i)_{i\in[c(\lambda)]}) = \mathsf{R}_{\mathsf{key}}$ and $(\mathsf{OPCt}, r) = ct$.
2. For indices $i \in [c(\lambda)]$ such that $(r)_i = 1$, apply $H^{\otimes\lambda}$ to R_i.
3. Run the program OPCt coherently on $(R_i)_{i\in[c(\lambda)]}$.
4. Measure the output register and output the outcome.

Theorem 16 ([16]). PKE *satisfies correctness and both CPA-style and random challenge message anti-piracy security.*

We claim that the construction is also LOCC-leakage-resilient.

Theorem 17. PKE *satisfies CPA-style LOCC leakage-resilience.*

When we instantiate the assumed building blocks with known constructions, we get the following corollary.

Corollary 3. *Assuming subexponentially secure $i\mathcal{O}$, one-way functions and polynomially hard qLWE, there exists a public-key encryption scheme that satisfies CPA-style LOCC leakage-resilience.*

Proof of Security. In this section, we give a proof sketch for Theorem 17 - were refer the reader to [15] for the full proof. We prove security through a series of hybrids, where we define Hyb_0 to be $\mathsf{PKE} - \mathsf{LOCC} - \mathsf{CPA}(\lambda, \mathcal{A})$.

Hyb_1*:* We now compute the challenge ciphertext, as follows, rather than as $ct \leftarrow \mathsf{PKE.Enc}(pk, m_b)$.

1. Sample $r \leftarrow \{0,1\}^{c(\lambda)}$.
2. Parse $(A_i, s_i, s'_i) = sk$.
3. Sample $r \leftarrow \{0,1\}^{c(\lambda)}$.
4. For $i \in [c(\lambda)]$, set $g_i = \mathsf{Can}_{A_i}$ if $(r)_i = 0$ and set $g_i = \mathsf{Can}_{(A_i)^\perp}$ if $(r)_i = 1$.
5. For $i \in [c(\lambda)]$, compute $y_i = g_i(s_i)$ if $(r)_i = 0$ and $y_i = g_i(s'_i)$ if $(r)_i = 1$.
6. Set g to be the function $g(v_1, \ldots, v_{c(\lambda)}) = (g_1(v_1)||\ldots||g_{c(\lambda)}(v_{c(\lambda)}))$.
7. Set $y = y_1||\ldots||y_{c(\lambda)}$.
8. Compute $\mathsf{OCC} \leftarrow \mathsf{CCObf.Obf}(g, y, m_b)$.
9. $\mathsf{OPCt} \leftarrow i\mathcal{O}(\mathsf{PCt}')$.

   $\mathsf{PCt}'(u_1, \ldots, u_{c(\lambda)})$
   Hardcoded: OCC
   (a) Output $\mathsf{OCC}(u_1, \ldots, u_{c(\lambda)})$.

10. Output (OPCt, r).

Hyb_2*:* We again change the computation of the challenge ciphertext by replacing lines 8 and 9 above with the following.

1. $\mathsf{PSim} \leftarrow \mathsf{CCObf.Sim}(1^\lambda, |g|, |y|, |m|)$
2. $\mathsf{OPCt} \leftarrow i\mathcal{O}(\mathsf{PCt}'')$.

   $\mathsf{PCt}''(u_1, \ldots, u_{c(\lambda)})$
   Hardcoded: PSim
   (a) Output $\mathsf{PSim}(u_1, \ldots, u_{c(\lambda)})$.

Claim. $\mathsf{Hyb}_0 \approx \mathsf{Hyb}_1$.

Proof. Note that $v \in A_i + s_i$ if and only if $\mathsf{Can}_{A_i}(v) = \mathsf{Can}_{A_i}(s_i)$. Similarly for $A_i^\perp + s'_i$. Then, by correctness of the inner obfuscations (i.e., $\mathsf{OP}_i^0, \mathsf{OP}_i^1$) in PCt and the correctness of CCObf, we have that PCt and PCt' have the same functionality. The result follows from the security of the outer obfuscation.

Claim. $\mathsf{Hyb}_1 \approx \mathsf{Hyb}_2$.

Proof. Suppose for a contradiction that there exists an adversary $\mathcal{A}$ such that $|\Pr[\mathsf{Hyb}_1 = 1] - \Pr[\mathsf{Hyb}_2 = 1]|$ is non-negligible. Then, we can show using compute-and-compare obfuscation security that there exists an adversary $\mathcal{A}''$ such that given the final state of the leakage adversary $\mathcal{A}$ and g, it outputs y with probability at least $2^{-\lambda^{0.5 \cdot C_{\mathsf{CompLOCC}}}}$.

Then, we construct an adversary $\mathcal{A}'''$ for $\mathsf{Coset} - \mathsf{CompLOCC}$ as follows.

$\underline{\mathcal{A}'''}$
Simulate $\mathcal{A}$ to obtain the leakage on the key and the challenge messages m_0, m_1, let ρ be the state of $\mathcal{A}$ at this point. Sample $b \leftarrow \{0,1\}$. Then, compute the description of g using the subspace descriptions (A_i, s_i, s_i') and r obtained from the challenger. Finally, run $\mathcal{A}''$ on (ρ, b, r) and g, and output the vectors output by it.

By above, we can see that $\mathcal{A}'''$ outputs vectors in the correct cosets with probability $2^{-\lambda^{0.5 \cdot C_{\mathsf{CompLOCC}}}}$ in the game $\mathsf{Coset} - \mathsf{CompLOCC}$, which is a contradiction by Theorem 15.

Observe that in Hyb_2, the challenge ciphertext is independent of b. Hence, $\Pr[\mathsf{Hyb}_2 = 1] \leq \frac{1}{2}$ and therefore $\Pr[\mathsf{PKE} - \mathsf{LOCC} - \mathsf{CPA}(\lambda, \mathcal{A})] \leq \frac{1}{2} + \mathsf{negl}(\lambda)$ by above.

6 Leakage-Resilient Quantum Bank

In this section, we introduce the notion of quantum banks (mints) with key protection and unbounded classical leakage-resilience for such schemes. Then, we give a construction relative to a classical oracle.

Definition 6 (Quantum Bank with Key Protection). *A quantum bank with key protection consists of the following QPT algorithms.*

- $\mathsf{Setup}(1^\lambda)$*: Outputs a public classical banknote verification key* vk *and a secret classical bank key* sk*.*
- $\mathsf{QKeyGen}(sk)$*: Takes as input the classical secret key* sk*, outputs a quantum key register* $\mathsf{R}_{\mathsf{bank}}$*.*
- $\mathsf{GenBanknote}(\mathsf{R}_{\mathsf{bank}}, m)$*: Takes as input the quantum bank key, outputs a quantum banknote.*
- $\mathsf{Ver}(vk, \mathsf{R}_{\mathsf{bn}})$*: Takes as input the public verification key and a (supposed) banknote* R_{bn}*, returns* 1 *if it is a valid banknote.*

We require that the scheme satisfies correctness, that is, any honestly generated banknote passes verification with probability 1*. We also require that each invocation of* $\mathsf{GenBanknote}$ *only negligibly disturbs the key* $\mathsf{R}_{\mathsf{bank}}$*, thus ensuring reusability for any polynomial number of times.*

We define counterfeiting security the same as the previous work [2]: any QPT adversary that obtains k banknotes cannot output $k+1$ banknotes except with negligible probability.

We now introduce the notion of LOCC leakage-resilience. We will require that a leakage adversary that has k banknotes and obtains leakage on the secret key of the bank will not be able to produce even a single extra banknote.

Definition 7 (LOCC Leakage-Resilience for Quantum Banks). *Let* Bank *be a quantum bank scheme with key protection. Consider the following game between the challenger and an LOCC leakage adversary* $\mathcal{A}$ *(Definition 4).*

$\underline{\mathsf{Bank} - \mathsf{LOCC}(\lambda, \mathcal{A})}$

1. *Sample* $vk, sk \leftarrow \mathsf{Bank.Setup}(1^\lambda)$.
2. *Sample* $\mathsf{R}_{\mathsf{bank}} \leftarrow \mathsf{Bank.QKeyGen}(sk)$ *and submit* vk *to* $\mathcal{A}$.
3. ***Banknote Query Phase:***[14] *For multiple rounds,* $\mathcal{A}$ *queries for a banknote. For each query, the challenger executes* $\mathsf{R}_{\mathsf{bn}} \leftarrow \mathsf{Bank.GenBanknote}(sk)$ *and submits* R_{bn} *to the adversary. Let* k *be the number of queries made by the adversary.*
4. ***Leakage Phase:*** $\mathcal{A}$ *obtains leakage on* $\mathsf{R}_{\mathsf{bank}}$.
5. $\mathcal{A}$ *outputs a* $(k+1)$*-partite register* $(\mathsf{R}_i)_{i\in[k+1]}$.
6. *The challenger tests* $\mathsf{Bank.Ver}(pk, \mathsf{R}_i)$ *for* $i \in [k+1]$. *It outputs* 1 *if all the tests pass; otherwise, it outputs* 0.

We say that the quantum bank scheme Bank *with key protection satisfies* LOCC leakage-resilience *if for any QPT LOCC leakage adversary* $\mathcal{A}$, $\Pr[\mathsf{Bank} - \mathsf{LOCC}(\lambda, \mathcal{A}) = 1] \leq \mathsf{negl}(\lambda)$.

We also define the weaker notion of unbounded leakage-resilience by restricting the adversary $\mathcal{A}$ to be an unbounded leakage adversary (Definition 4).

6.1 Unbounded Leakage-Resilient Scheme Based on Classical Oracles

In this section, we give an unbounded leakage-resilient quantum bank scheme, using in a black-box way a quantum lightning scheme [30], which has a candidate construction relative to a classical oracle [7].

The construction of our chosen message secure leakage-resilient signature scheme[15] is almost the same as our quantum bank construction, with the difference being the signature scheme additionally has a classical signature on the message.

Let QL be a quantum lightning scheme.

$\underline{\mathsf{Bank.Setup}(1^\lambda)}$

1. Sample a subspace $A \subseteq \mathbb{F}_2^\lambda$ of dimension $\lambda/2$.
2. Sample random functions[16] H_1, H_2 with domain $\{0,1\}^{m(\lambda)}$ and output space $\{0,1\}^{p(\lambda)}$.
3. Construct the following oracles.

[14] Note that banknote queries being before the leakage phase is without loss of generality since there is no input for the banknote queries.

[15] See the full version [15] for the construction and security proof.

[16] If we insist on efficiency of the oracles, we can instead use PRFs.

$\mathcal{O}_1(v, sn)$ **Hardcoded:** A, H_1 (a) Check if $v \in A$. (b) If so, output $H_1(sn)$. Otherwise, output $\perp$.
$\mathcal{O}_2(v, sn)$ **Hardcoded:** A, H_2 (a) Check if $v \in A^{\perp}$. (b) If so, output $H_2(sn)$. Otherwise, output $\perp$.
$\mathcal{O}_3(sn, y)$ **Hardcoded:** H_1 (a) Output 1 if $H_1(sn) = y$, otherwise output 0.
$\mathcal{O}_4(sn, y)$ **Hardcoded:** H_2 (a) Output 1 if $H_2(sn) = y$, otherwise output 0.

4. Set $vk = (\mathcal{O}_1, \mathcal{O}_2, \mathcal{O}_3, \mathcal{O}_4)$ and $sk = A$.
5. Output (vk, sk).

$\mathsf{Bank.QKeyGen}(sk)$

1. Parse $A = sk$ and output $|A\rangle$.

$\mathsf{Bank.GenBanknote}(\mathsf{R}_{\mathsf{key}})$

1. Sample $\mathsf{R}_{\mathsf{bolt}}, sn \leftarrow \mathsf{QL.Bolt}(1^{\lambda})$.
2. Query $\mathcal{O}_1$ coherently on $\mathsf{R}_{\mathsf{key}}$ and sn and measure the result to obtain y_1; then rewind.
3. Query $\mathcal{O}_2$ coherently on $\mathsf{R}_{\mathsf{key}}$ and sn and measure the result to obtain y_2; then rewind.
4. Output $\mathsf{R}_{\mathsf{bolt}}, sn, y_1, y_2$.

$\mathsf{Bank.Ver}(vk, \mathsf{R})$

1. Parse $(\mathsf{R}_{\mathsf{bolt}}, sn, y_1, y_2) = \mathsf{R}$.
2. Verify $\mathsf{QL.Ver}(sn, \mathsf{R}_{\mathsf{bolt}}) = 1$.
3. Check if $\mathcal{O}_3(sn, y_1) = 1$ and $\mathcal{O}_4(sn, y_2) = 1$.
4. Output 1 if all the checks pass; otherwise, output 0.

Theorem 18. Bank *satisfies correctness, reusability and unbounded leakage-resilience.*

Correctness and reusability are easy to see by the correctness of the lightning scheme and As Good As New Lemma [1]. We refer the reader to the full version [15] for the leakage-resilience proof.

7 Relationship Between Unclonability and Leakage-Resilience

In this section, we show a public-key encryption scheme that satisfies anti-piracy security (and hence unbounded leakage-resilience) but not even 1-round LOCC leakage-resilience.

Theorem 19. *Assuming equivocal CRHF Definition 2, relative to a classical oracle, there exists a public-key encryption scheme that satisfies anti-piracy security (both CPA-style and random-challenge) but there exists a 1-round LOCC leakage adversary that wins the leakage-resilience game against it with overwhelming probability.*

We note by an argument similar to the proof of this theorem, the same result can be obtained for signature schemes and PRFs.

We now give our construction, based on equivocal CRHF, which has a candidate construction relative to a classical oracle [7]. Let P be the partitioning, H_P be the associated hash function and $\mathcal{O}_P, \mathcal{O}_P^{\perp}$ be the associated oracles, all as in Theorem 11. Set $\ell(\lambda) = \lambda/2$, the output length of the hash function H_P.

$\underline{\mathsf{PKE.Setup}(1^\lambda)}$

1. Sample subspaces $A_i \subseteq \mathbb{F}_2^\lambda$ of dimension $\lambda/2$ for $i \in [\ell]$. Sample random functions $F_1 : \{0,1\}^\lambda \to \{0,1\}^{2\lambda}$, $F_2 : \{0,1\}^{2\lambda} \to \{0,1\}^{m(\lambda)}$, $F_3 : \{0,1\}^{2\lambda} \to \{0,1\}^{m(\lambda)}$.
2. Construct the following oracles.

$\underline{\mathcal{O}_1(x, y, (v_i)_{i\in[\ell]}, r)}$
Hardcoded: $(A_i)_{i\in[\ell]}, F_2, H_P$
(a) For $i \in [\ell]$, check if $v_i \in A_i$ if $(y)_i = 0$ and check if $v_i \in A_i^{\perp}$ if $(y)_i = 1$.
(b) Check if $(x)_1 = 0$.
(c) Check if $H_P(x) = y$.
(d) If any of the checks above failed, output $\perp$. Otherwise, output $F_2(r)$.

$\underline{\mathcal{O}_2(x, y, (v_i)_{i\in[\ell]}, r)}$
Hardcoded: $(A_i)_{i\in[\ell]}, F_3, H_P$
(a) For $i \in [\ell]$, check if $v_i \in A_i$ if $(y)_i = 0$ and check if $v_i \in A_i^{\perp}$ if $(y)_i = 1$.
(b) Check if $(x)_1 = 1$.
(c) Check if $H_P(x) = y$.
(d) If any of the checks above failed, output $\perp$. Otherwise, output $F_3(r)$.

$\underline{\mathcal{O}_3(s)}$
Hardcoded: F_1, F_2, F_3
(a) $r = F_1(s)$.
(b) Output $r, F_2(r), F_3(r)$.

3. Set $pk = (\mathcal{O}_1, \mathcal{O}_2, \mathcal{O}_3, \mathcal{O}_P, \mathcal{O}_P^{\perp})$ and $sk = ((A_i)_{i\in[\ell]})$.
4. Output (pk, sk).

$\underline{\mathsf{PKE.QKeyGen}(sk)}$

1. Parse $(A_i)_{i\in[\ell]} = sk$ and output $|A_i\rangle_{i\in[\ell]}$.

$\underline{\mathsf{PKE.Enc}(pk, m)}$

1. Sample $s \leftarrow \{0,1\}^{\lambda}$.
2. Query $\mathcal{O}_3(s)$ to get r, r_2, r_3.
3. Output $r, r_2 \oplus r_3 \oplus m$.

$\underline{\mathsf{PKE.Dec}(\mathsf{R}_{\mathsf{key}}, ct)}$

1. Parse $|A_i\rangle_{i\in[\ell]} = \mathsf{R}_{\mathsf{key}}$.
2. Parse $r, z = ct$.
3. Execute $\mathsf{ECRHF.Gen}^{\mathcal{O}_P, \mathcal{O}_P^{\perp}}$ to get $|\psi_y\rangle, y$.
4. Run $\mathsf{Equiv}^{\mathcal{O}_P, \mathcal{O}_P^{\perp}}(|\psi_y\rangle, 0)$ to obtain x^0. Query $\mathcal{O}_1$ coherently on x^0, y, $|A_i\rangle_{i\in[\ell]}, r$ to obtain r_2. Then, rewind the quantum states back to $|\psi_y\rangle$ and $|A_i\rangle_{i\in[\ell]}$.
5. Run $\mathsf{Equiv}^{\mathcal{O}_P, \mathcal{O}_P^{\perp}}(|\psi_y\rangle, 1)$ to obtain x^1. Query $\mathcal{O}_2$ coherently on x^1, y, $|A_i\rangle_{i\in[\ell]}, r$ to obtain r_3. Then, rewind the quantum states back to $|\psi_y\rangle$ and $|A_i\rangle_{i\in[\ell]}$.
6. Output $z \oplus r_2 \oplus r_3$.

Proof of Correctness and 1-Round LOCC Attack. Correctness with overwhelming probability follows in a straightforward way from the correctness of the equivocal CRHF and by As Good As New Lemma [1]. We give the following 1-round LOCC attack. The adversary $\mathcal{A}$ executes $\mathsf{ECRHF.Gen}^{\mathcal{O}_P, \mathcal{O}_P^{\perp}}$ to get $|\psi_y\rangle, y$. Then, it chooses a measurement that measures each part (subspace state) of the key (subspace state tuple) either in the computational basis (if $(y)_i = 0$) or in the Hadamard basis (if $(y)_i = 1$). The resulting leakage consists of vectors $(v_i)_{i\in[\ell]}$ such that $v_i \in A_i$ if $(y)_i = 0$ and $v_i \in A_i^{\perp}$ if $(y)_i = 1$. Then, $\mathcal{A}$ uses $|\psi_y\rangle, y$ and the vectors $(v_i)_{i\in[\ell]}$ to simulate $\mathsf{PKE.Dec}$ from Step 4 onwards. It is easy to see that this perfectly simulates $\mathsf{PKE.Dec}$. Thus, $\mathcal{A}$ wins with overwhelming probability.

Proof of Anti-piracy Security. In this section, we give a proof sketch that PKE satisfies anti-piracy security. We refer the reader to the full version [15] for the relevant definitions, notation and the full proof. We use *projective and threshold implementations* along with their approximate versions [3,30] in our proof to extract subspace vectors from entangled freeloaders. Throughout the proof, for simplicity we will use (inefficient) threshold implementations – however, the

same result follows with the efficient approximate versions [15]. Further, we in fact prove that the scheme satisfies *strong γ-anti-piracy* [14,16] for any inverse polynomial, which implies it also satisfies regular anti-piracy security [16].

Let γ be any inverse polynomial, and suppose there exists an anti-piracy adversary that wins the strong γ-anti-piracy with non-negligible probability $1/q(\lambda)$. Let τ be the bipartite state output by the adversary $\mathcal{A}$ for the freeloaders. Throughout the proof, we will assume that the first parts of the challenge ciphertexts are new random values (rather than outputs of F_1) – since F_1 is a random function, this does not change the behavior of the freeloaders. Further, we will implicitly assume that the freeloaders have access to a modified oracle $\mathcal{O}_3'$ that has the functions F_2, F_3 inside punctured at r_1^*, r_2^*, the first part of the challenge ciphertexts. It is easy to see that with overwhelming probability this does not change the success probabilities of any decryptor/freeloader, since with overwhelming probability these values r_1^*, r_2^* will be outside the image set of F_1. Now, apply the tests $\mathsf{TI}_{1,\mathcal{D},1/2+\gamma} \otimes \mathsf{TI}_{2,\mathcal{D},1/2+\gamma}$ to τ, and condition on both parts accepting (which happens with probability $1/q(\lambda)$) and let τ' denote the conditioned post-measurement state. We note that, since TI are projective measurements, we now know that each part $\ell \in \{1,2\}$ of τ' will again pass if it is tested with $\mathsf{TI}_{\ell,\mathcal{D},1/2+\gamma}$, even if we condition on some non-negligible probability event on the other side.

First, we claim that we can extract subspace vectors from the first freeloader, $\tau'[1]$.

Lemma 1. *Define the test $\mathsf{TI}'_{1,\mathcal{D},1/2+\gamma}$ to be the same as $\mathsf{TI}_{1,\mathcal{D},1/2+\gamma}$ except that the freeloader is now given access to an empty oracle instead of $\mathcal{O}_1$. Then, applying $\mathsf{TI}'_{1,\mathcal{D},1/2+\gamma}$ to any state σ accepts with negligible probability.*

Proof. Observe that in this case (i.e., with no access to $\mathcal{O}_1$), given the view of the freeloader adversary, the value $F_2(r_1^*)$ is truly random. Thus, the ciphertext $m \oplus F_2(r_1^*) \oplus F_3(r_2^*)$ is a perfect one-time pad encryption, and no adversary can succeed with probability $1/2 + \gamma$.

Lemma 2. *Define the test $\mathsf{TI}''_{2,\mathcal{D},1/2+\gamma}$ to be the same as $\mathsf{TI}_{2,\mathcal{D},1/2+\gamma}$ except that the freeloader is now given access to an empty oracle instead of $\mathcal{O}_2$. Then, applying $\mathsf{TI}'_{1,\mathcal{D},1/2+\gamma}$ to any state σ accepts with negligible probability.*

Proof. Follows from the same argument as above.

Corollary 4. *Simulate the test $\mathsf{ATI}_{1,\mathcal{D},1/2+\gamma}$ on $\tau'[1]$ and measure a random query to $\mathcal{O}_1$. Condition on obtaining an outcome $x, y, (v_i)_{i\in[\ell]}$ such that $(x)_1 = 0, H(x) = y$ and for all $i \in [\ell]$, $v_i \in A_i$ if $(y)_i = 0$ and $v_i \in A_i^\perp$ if $(y)_i = 1$. Next, simulate the test $\mathsf{ATI}_{1,\mathcal{D},1/2+\gamma}$ on the second freeloader of the conditioned state and measure a random query to $\mathcal{O}_2$. Then, with non-negligible probability, we get values $x', y', (w_i)_{i\in[\ell]}$ such that $(x')_1 = 1, H(x') = y'$ and for all $i \in [\ell]$, $w_i \in A_i$ if $(y')_i = 0$ and $w_i \in A_i^\perp$ if $(y')_i = 1$; and finally $y \neq y'$.*

Proof. Since $\tau'[1]$ passes $\mathsf{TI}_{1,\mathcal{D},1/2+\gamma}$ with non-negligible probability but passes $\mathsf{TI}'_{1,\mathcal{D},1/2+\gamma}$ with negligible probability, we get the first part by the query weight

lemma [12]. Now, as argued above, we know that even conditioned on such an extraction (which happens with non-negligible probability) from the first freeloader/register, the second part still passes $\mathsf{TI}_{2,\mathcal{D},1/2+\gamma}$ with non-negligible probability. However, by the above lemma we know that it passes $\mathsf{TI}''_{2,\mathcal{D},1/2+\gamma}$ with negligible probability. Thus, we get second part again by [12]. Finally, we argue the third part ($y \neq y'$) as follows. Observe that if $y = y'$, we have $H(x) = H(x')$ but $(x)_1 = 0$ and $(x')_1 = 1$, meaning that $x \neq x'$. Thus, this gives us a collision for the hash function H, which is a contradiction.

Observe that the above gives us a way of breaking the direct product hardness [11] of the subspace state at the index where y and y' differ. Thus, we arrive at a contradiction, concluding the security proof.

Acknowledgments. Part of the work done while C.L. was at NTT Research and J.R. was at NOVA LINCS, NOVA School of Science and Technology, and Carnegie Mellon University. A.C. was supported by the following grants of V.G.: NSF award 1916939, DARPA SIEVE program, a gift from Ripple, a DoE, NETL award, a JP Morgan Faculty Fellowship, a PNC center for financial services innovation award, and a Cylab seed funding award. J.R. was partially supported by the same grants and also by NOVA LINCS (UIDB/04516/2020) with the financial support of FCT.IP. C.L. was partially supported by the Hasler Foundation Project no 23090.

References

1. Aaronson, S.: The complexity of quantum states and transformations: from quantum money to black holes. arXiv preprint arXiv:1607.05256 (2016)
2. Aaronson, S., Christiano, P.: Quantum money from hidden subspaces. In: Proceedings of the Forty-Fourth Annual ACM Symposium on Theory of Computing, pp. 41–60 (2012)
3. Aaronson, S., Liu, J., Liu, Q., Zhandry, M., Zhang, R.: New approaches for quantum copy-protection. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021. LNCS, vol. 12825, pp. 526–555. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-84242-0_19
4. Agrawal, D., Archambeault, B., Rao, J.R., Rohatgi, P.: The EM side–channel(s). In: Kaliski, B.S., Koç, Ç.K., Paar, C. (eds.) Cryptographic Hardware and Embedded Systems - CHES 2002, pp. 29–45. Springer, Heidelberg (2003)
5. Agrawal, S., Kitagawa, F., Nishimaki, R., Yamada, S., Yamakawa, T.: Public key encryption with secure key leasing. In: Hazay, C., Stam, M. (eds.) Advances in Cryptology - EUROCRYPT 2023, pp. 581–610. Springer, Cham (2023)
6. Ahmad, A., Lee, S., Peinado, M.: Hardlog: Practical tamper-proof system auditing using a novel audit device. In: 2022 IEEE Symposium on Security and Privacy (SP), pp. 1791–1807 (2022). https://doi.org/10.1109/SP46214.2022.9833745
7. Amos, R., Georgiou, M., Kiayias, A., Zhandry, M.: One-shot signatures and applications to hybrid quantum/classical authentication. In: Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, pp. 255–268 (2020)
8. Ananth, P., La Placa, R.L.: Secure software leasing. In: Canteaut, A., Standaert, F.-X. (eds.) EUROCRYPT 2021. LNCS, vol. 12697, pp. 501–530. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-77886-6_17

9. Bartusek, J., Goyal, V., Khurana, D., Malavolta, G., Raizes, J., Roberts, B.: Software with certified deletion. In: Joye, M., Leander, G. (eds.) Advances in Cryptology - EUROCRYPT 2024, pp. 85–111. Springer, Cham (2024)
10. Bartusek, J., Khurana, D.: Cryptography with certified deletion. In: Handschuh, H., Lysyanskaya, A. (eds.) Advances in Cryptology – CRYPTO 2023, pp. 192–223 (2023)
11. Ben-David, S., Sattath, O.: Quantum tokens for digital signatures. Quantum **7**, 901 (2023). https://doi.org/10.22331/q-2023-01-19-901
12. Bennett, C.H., Bernstein, E., Brassard, G., Vazirani, U.: Strengths and weaknesses of quantum computing. SIAM J. Comput. **26**(5), 1510–1523 (1997)
13. Brian, G., et al.: The mother of all leakages: how to simulate noisy leakages via bounded leakage (almost) for free. IEEE Trans. Inf. Theory **68**(12), 8197–8227 (2022). https://doi.org/10.1109/TIT.2022.3193848. Preliminary version in Eurocrypt 2021
14. Çakan, A., Goyal, V.: Unclonable cryptography with unbounded collusions (2023). https://eprint.iacr.org/2023/1841
15. Cakan, A., Goyal, V., Liu-Zhang, C.D., Ribeiro, J.: Unbounded leakage-resilience and intrusion-detection in a quantum world. Cryptology ePrint Archive, Paper 2023/410 (2023). https://eprint.iacr.org/2023/410
16. Coladangelo, A., Liu, J., Liu, Q., Zhandry, M.: Hidden cosets and applications to unclonable cryptography. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021. LNCS, vol. 12825, pp. 556–584. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-84242-0_20
17. Kalai, Y.T., Reyzin, L.: A survey of leakage-resilient cryptography. In: Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali, pp. 727–794. ACM (2019)
18. Kitagawa, F., Nishimaki, R.: Functional encryption with secure key leasing. In: Agrawal, S., Lin, D. (eds.) Advances in Cryptology - ASIACRYPT 2022, pp. 569–598. Springer, Cham (2022)
19. Kocher, P.C.: Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In: Koblitz, N. (ed.) Advances in Cryptology — CRYPTO 1996, pp. 104–113 (1996)
20. Liu, J., Liu, Q., Qian, L., Zhandry, M.: Collusion resistant copy-protection for watermarkable functionalities. In: Kiltz, E., Vaikuntanathan, V. (eds.) Theory of Cryptography, TCC 2022, pp. 294–323. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-22318-1_11
21. Pass, R., Shelat, A.: A course in cryptography (2010)
22. Quisquater, J.J., Samyde, D.: ElectroMagnetic analysis (EMA): measures and counter-measures for smart cards. In: Attali, I., Jensen, T. (eds.) Smart Card Programming and Security, pp. 200–210. Springer, Heidelberg (2001)
23. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. J. ACM **56**(6), 1–40 (2009)
24. Sahai, A., Waters, B.: How to use indistinguishability obfuscation: deniable encryption, and more. In: Proceedings of the Forty-Sixth Annual ACM Symposium on Theory of Computing (STOC 2014), pp. 475–484. Association for Computing Machinery, New York (2014). https://doi.org/10.1145/2591796.2591825
25. Sinha, A., Jia, L., England, P., Lorch, J.R.: Continuous tamper-proof logging using TPM 2.0. In: Holz, T., Ioannidis, S. (eds.) Trust and Trustworthy Computing, pp. 19–36. Springer, Cham (2014)

26. Snodgrass, R.T., Yao, S.S., Collberg, C.: Tamper detection in audit logs. In: Proceedings of the Thirtieth International Conference on Very Large Data Bases (VLDB 2004), vol. 30, pp. 504–515. VLDB Endowment (2004)
27. Vidick, T., Zhang, T.: Classical proofs of quantum knowledge. In: Canteaut, A., Standaert, F.-X. (eds.) EUROCRYPT 2021. LNCS, vol. 12697, pp. 630–660. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-77886-6_22
28. Wichs, D., Zirdelis, G.: Obfuscating compute-and-compare programs under LWE. In: 2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS), pp. 600–611 (2017). https://doi.org/10.1109/FOCS.2017.61
29. Zhandry, M.: Quantum lightning never strikes the same state twice. In: Ishai, Y., Rijmen, V. (eds.) EUROCRYPT 2019. LNCS, vol. 11478, pp. 408–438. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-17659-4_14
30. Zhandry, M.: Schrödinger's pirate: how to trace a quantum decoder. In: Pass, R., Pietrzak, K. (eds.) Theory of Cryptography, pp. 61–91. Springer, Cham (2020)

Math & Foundations

Bit-Security Preserving Hardness Amplification

Shun Watanabe[1(✉)] and Kenji Yasunaga[2]

[1] Tokyo University of Agriculture and Technology, Tokyo, Japan
shunwata@cc.tuat.ac.jp
[2] Institute of Science Tokyo, Tokyo, Japan
yasunaga@c.titech.ac.jp

Abstract. Hardness amplification is one of the important reduction techniques in cryptography, and it has been extensively studied in the literature. The standard XOR lemma known in the literature evaluates the hardness in terms of the probability of correct prediction; the hardness is amplified from mildly hard (close to 1) to very hard $1/2+\varepsilon$ by inducing ε^2 multiplicative decrease of the circuit size. Translating such a statement in terms of the bit-security framework introduced by Micciancio-Walter (EUROCRYPT 2018) and Watanabe-Yasunaga (ASIACRYPT 2021), it may cause a bit-security loss of $\log(1/\varepsilon)$. To resolve this issue, we derive a new variant of the XOR lemma in terms of the Rényi advantage, which directly characterizes the bit security. In the course of proving this result, we prove a new variant of the hardcore lemma in terms of the conditional squared advantage; our proof uses a boosting algorithm that may output the $\perp$ symbol in addition to 0 and 1, which may be of independent interest.

1 Introduction

In modern cryptography, cryptographic primitives are usually proposed with security proofs. When proving the security of a primitive under some hardness assumption, we show a *reduction* that solves a hard problem by assuming the existence of an adversary attacking the primitive. If the reduction requires much more computational cost than the assumed adversary, we need a stronger hardness assumption to achieve a target level of security. Thus, *tight* reductions of security proofs are desirable for the efficient use of cryptographic primitives.

A recent approach of *concrete security* reveals quantities related to security reductions. Suppose we want to prove the security of primitive Q by assuming the security of primitive P (or the hardness of some problem). Typically, we show that for any adversary B of primitive Q with running time $t_B(n)$ and advantage $\varepsilon_B(n)$, there is an adversary A of primitive P such that the running time $t_A(n)$ and the advantage $\varepsilon_A(n)$ satisfy $t_A(n) \leq \phi(t_B(n))$ and $\varepsilon_A(n) \geq \psi(\varepsilon_B(n))$ for some functions ϕ and ψ. Here, n is a security parameter, and a reduction is a construction of A out of B. We may understand the tightness

E. Boyle and M. Mahmoody (Eds.): TCC 2024, LNCS 15365, pp. 195–223, 2025.
https://doi.org/10.1007/978-3-031-78017-2_7

of the reduction by specifying two functions, ϕ and ψ. We prefer smaller $t_A(n)$ and larger $\varepsilon_A(n)$ for tight reductions. Thus, it is tempted to combine the two quantities as $t_A(n)/\varepsilon_A(n)$ and achieve the value as small as $t_B(n)/\varepsilon_B(n)$ of adversary B. Namely, we want the loss function

$$L(n) = \frac{t_A(n)}{\varepsilon_A(n)} \cdot \frac{\varepsilon_B(n)}{t_B(n)}$$

to be as small as possible.

The above way of quantifying security loss has been used in cryptographic literature. In [29], the quantity of $t_A(n)/\varepsilon_A(n)$ was used to define the security of primitives. The same treatment has been employed in the literature [2,4,6, 33]. For *search* primitives such as one-way functions and signature schemes, the advantage $\varepsilon_A(n)$ is simply defined as the adversary's success probability. For *decision* primitives such as pseudorandom generators and encryption schemes, it is usually defined as the gap between two probabilities, which we want to make as small as possible. This treatment of defining advantages has been standard in the cryptography community. In the literature listed above, the advantage $\varepsilon_A(n)$ was defined in this way for analyzing the quantity $t_A(n)/\varepsilon_A(n)$.

In [12], Goldreich noted that Levin suggested using another quantity $t_A(n)/\varepsilon_A^2(n)$, called *work*, for decision primitives. The reason is that if the gap of two probabilities is $\varepsilon_A(n)$, we need to repeat the experiment (security game) for $\mathcal{O}(1/\varepsilon_A(n)^2)$ times to amplify it to a constant, say 2/3. This was also suggested in [18]. However, the use of this quantity was not justified well at that time.

1.1 Bit Security

Micciancio and Walter [32] initiated a theoretical study for quantifying the security level of primitives, referring to it as *bit security*. They proposed using another notion of advantage, which we call *conditional squared (CS) advantage*, for evaluating the decision primitives. It is defined as

$$\mathsf{Adv}_A^{\mathrm{CS}}(n) = \Pr(Y \neq \perp)(2\Pr(Y = U | Y \neq \perp) - 1)^2,$$

where Y is the random variable representing the adversary A's output and U is a (randomly chosen) secret bit of the decision game. They defined the bit security as the quantity of

$$\min_A \log_2 \frac{t_A(n)}{\mathsf{Adv}_A^{\mathrm{CS}}(n)}.$$

Their notion elegantly resolved paradoxical situations in pseudorandom generators and approximate samplers (For the former case, we elaborate later in this section). In [28], the notion of [32] was extended for capturing both computational and statistical parameters. Watanabe and Yasunaga [39] defined bit security with an *operational meaning* to justify formalizing the security level of primitives. Roughly speaking, for a given adversary of a primitive, the bit security in [39] is defined as the number of invocations of that adversary to break

the primitive with probability close to 1; it was shown in [39] that the *Rényi advantage* characterizes the number of invocations. Specifically, it is given by

$$\mathsf{Adv}_A^{\mathrm{Renyi}}(n) = -2\ln \sum_a \sqrt{\Pr(Y=a|U=0)\Pr(Y=a|U=1)},$$

(See Appendix A for more explanations of the two frameworks [32,39].) The follow-up work [40] demonstrated that the two advantages of [32,39] are essentially equivalent;[1] owing to this result, we can use the CS advantage and the Rényi advantage almost interchangeably, and we will use both the advantages in this paper as well.

1.2 Hardness Amplification

In this work, based on the recent advances in the notion of bit security (or the quantity $t_A(n)/\varepsilon_A(n)$), we focus on a basic problem of *hardness amplification* [14] of Boolean functions. A Boolean function $f : \{0,1\}^n \to \{0,1\}$ is said to be *mildly hard (unpredictable)* if every polynomial-time algorithm fails to compute f on a δ-fraction of input $x \in \{0,1\}^n$ for a noticeable δ. The task of hardness amplification is to convert f into another function f' so that f' is *strongly hard* in the sense that every polynomial-time algorithm fails to compute f' on a $(1/2-\varepsilon)$-fraction of input. The most well-known technique is Yao's XOR lemma; $f'(x_1,\ldots,x_k) = f(x_1)\oplus\cdots\oplus f(x_k)$ for $x_i \in \{0,1\}^n$. In cryptography, Yao's XOR lemma (and its variants) have been used for amplifying the security of pseudorandom generators (PRGs) and pseudorandom functions (PRFs) [7,31,34]. In the framework of bit security, hardness amplification is to reduce the advantage $\mathsf{Adv}_{A,f}(n)(=1/2-\delta)$ to $\mathsf{Adv}_{B,f'}(n)(=\varepsilon)$, where $\mathsf{Adv}_{A,f}(n)$ is the advantage of adversary A predicting f over random guessing.

In the two bit-security frameworks of [32,39], a decision game is formalized such that an adversary tries to guess the secret bit $u \in \{0,1\}$ by playing the game. Thus, we can write a decision game as $G = (G_0, G_1)$, where a secret bit u is initially chosen uniformly at random, and the adversary plays G_u for guessing u. The hardness of predicting a Boolean function $f : \{0,1\}^n \to \{0,1\}$ can be captured by the game G_f as follows; first, $x \in \{0,1\}^n$ and $\sigma \in \{0,1\}$ are chosen uniformly at random. Then, the adversary receives $(x, f(x))$ when $u = 0$, and (x,σ) when $u = 1$, and outputs a symbol in $\{0,1,\perp\}$.[2]

When employing the framework of [39,40], the bit security of game $G_f = (G_0, G_1)$ against adversaries with computational cost s can be approximated as

$$\mathrm{BS}_s(G_f) = \log \min_{A \text{ with cost } s} \frac{s}{\mathsf{Adv}_{A,G_f}^{\mathrm{Renyi}}}, \tag{1}$$

[1] Generally, the CS advantage is bounded above by the Rényi advantage. While the CS advantage may take a much smaller value in some cases, the CS advantage can be increased to the level of the Rényi advantage by modifying adversaries appropriately.

[2] The symbol $\perp$ indicates that the adversary gives up the prediction.

where $\mathsf{Adv}^{\mathrm{Renyi}}_{A,G_f} = D_{1/2}(A_0 \| A_1)$ is the Rényi advantage of A in game G_f, $D_{1/2}(\cdot\|\cdot)$ is the Rényi divergence of order $1/2$, and A_u is the output distribution of A when playing G_u.

With the notions of bit security, hardness amplification is the task of converting f into f' such that $\max_B \mathsf{Adv}^{\mathrm{Renyi}}_{B,G_{f'}}$ is much smaller than $\max_A \mathsf{Adv}^{\mathrm{Renyi}}_{A,G_f}$, where A and B are taken over adversaries with costs s and s', respectively. We want the following loss function

$$L^{\mathrm{amp}}(n) = \frac{s \cdot \max_B \mathsf{Adv}^{\mathrm{Renyi}}_{B,G_{f'}}}{s' \cdot \max_A \mathsf{Adv}^{\mathrm{Renyi}}_{A,G_f}}$$

to be as small as possible. Ideally, we want to achieve $L^{\mathrm{amp}}(n) = \mathcal{O}(1)$.

Balanced/Unbalanced Adversaries. The most efficient reductions of the XOR lemma until now were given in [3,25] using boosting versions of *hardcore lemmas* [23]. They guarantee $s' = \Omega(\varepsilon^2/\log(1/\delta)) \cdot s$, where $\max_A \Pr(A(x) = f(x)) = 1 - \delta$ and $\max_B \Pr(B(x) = f'(x)) = 1/2 + \varepsilon$, where the maxima are taken over algorithms with cost s and s', respectively. Such a *predictor* A with $\Pr(A(x) = f(x)) = 1 - \delta$ can be easily converted to a *distinguisher* A' with the same cost such that $\mathsf{Adv}^{\mathrm{TV}}_{A',G_f} = d_{\mathsf{TV}}(A'_0, A'_1) = (1-\delta) - 1/2 = 1/2 - \delta$, where $d_{\mathsf{TV}}(\cdot,\cdot)$ is the total variation distance. For any adversary A of game $G_f = (G_0, G_1)$, it holds that

$$\left(\mathsf{Adv}^{\mathrm{TV}}_{A,G_f}\right)^2 \le \mathsf{Adv}^{\mathrm{Renyi}}_{A,G_f} \le \mathcal{O}\left(\mathsf{Adv}^{\mathrm{TV}}_{A,G_f}\right).$$

Thus, if $\mathsf{Adv}^{\mathrm{Renyi}}_{A,G_f} \approx \left(\mathsf{Adv}^{\mathrm{TV}}_{A,G_f}\right)^2$ holds for every adversary A, the XOR lemma in [3,25] gives

$$L^{\mathrm{amp}}(n) = \mathcal{O}\left(\frac{\log(1/\delta)}{\varepsilon^2} \cdot \frac{\varepsilon^2}{(1/2-\delta)^2}\right) = \mathcal{O}(\log(1/\delta)),$$

meaning that the reduction seems to be optimal. Indeed, as observed in [39], $\mathsf{Adv}^{\mathrm{Renyi}}_{A,G_f} \approx \left(\mathsf{Adv}^{\mathrm{TV}}_{A,G_f}\right)^2$ holds for *balanced* adversaries, who output every value with probability $\Omega(1)$. However, generally, we have $\mathsf{Adv}^{\mathrm{Renyi}}_{A,G_f} = \mathcal{O}\left(\mathsf{Adv}^{\mathrm{TV}}_{A,G_f}\right)$. Thus, the reductions in [3,25] imply that

$$L^{\mathrm{amp}}(n) = \mathcal{O}\left(\frac{\log(1/\delta)}{\varepsilon^2} \cdot \frac{\varepsilon}{1/2-\delta}\right) = \mathcal{O}\left(\frac{\log(1/\delta)}{\varepsilon}\right), \tag{2}$$

which does not seem to be optimal.

It should be emphasized that the above-mentioned difference between the balanced and unbalanced adversaries is crucial in the bit security frameworks of [32,39]; the difference can be well explained by the following example first posed in [32] and further elaborated in [40]. Let us consider the decision game

to distinguish between the pseudorandom number generator (PRG) and the true random number generator (TRG): the outcome (y, z) of PRG consists of the image $y = F(r)$ of a one-way permutation F over $\{0,1\}^n$ and its hard-core predicate $z = h(r)$; the outcome (y, z) of TRG consists of $y = F(r)$ and a random bit $z = \sigma$ that is independent of the seed r. For this game, we can consider the following two possible attacks:

1. *Linear test attack:* For a prescribed binary vector v of length $n + 1$, the adversary computes the inner product of v and (y, z); if the outcome is 0, the adversary outputs 0 (PRG); and outputs 1 (TRG) otherwise. For such an attack, the output distribution A_u of the adversary A given $u \in \{0,1\}$ ($u = 0$ represent PRG and $u = 1$ represent TRG) are $A_0 = (1/2 + \varepsilon, 1/2 - \varepsilon)$ and $A_1 = (1/2, 1/2)$ for some bias ε, where $A_u = (p_0, p_1)$ means that $\Pr[A = 0 \mid u] = p_0$ and $\Pr[A = 1 \mid u] = p_1$.
2. *Inversion attack:* First, the adversary tries to invert the one-way permutation, which will succeed with probability ε. If the inversion is unsuccessful, the adversary outputs $\perp$; if the inversion is successful and $h(r)$ coincides with z, then the adversary outputs 0 (PRG); otherwise (if the inversion is successful but $h(r) \neq z$), then the adversary outputs 1 (TRG). For such an attack, the output distribution of the adversary A given $u \in \{0,1\}$ consists of $A_0 = (\varepsilon, 0, 1 - \varepsilon)$ and $A_1 = (\varepsilon/2, \varepsilon/2, 1 - \varepsilon)$, where the third coordinate represents the probability that $\Pr[A = \perp \mid u]$.

The standard advantage $\mathsf{Adv}^{\mathrm{TV}}$, the CS advantage $\mathsf{Adv}^{\mathrm{CS}}$, and the Rényi advantage for the linear test attack and the inversion attack are summarized in Table 1. Note that, in the inversion attack, the adversary outputs $\perp$ with significant probability $1-\varepsilon$, and it has a completely different nature compared to the linear test attack. However, the standard advantages of the two attacks are almost the same, and it does not capture the different nature of the attacks. One of the notable features of the bit security in [32,39] is that the difference between these attacks is reflected as different orders of the advantages. Thus, it is necessary to extend the hardness amplification to the CS advantage or the Rényi advantage, which involves several challenges mentioned in the next section.

Table 1. Advantages for the linear test attack and the inversion attack

Attacks	$\mathsf{Adv}^{\mathrm{TV}}$	$\mathsf{Adv}^{\mathrm{CS}}$	$\mathsf{Adv}^{\mathrm{Renyi}}$
Linear test attack (balanced) $A_0 = (1/2 + \varepsilon, 1/2 - \varepsilon)$ $A_1 = (1/2, 1/2)$	ε	ε^2	$\Theta(\varepsilon^2)$
Inversion attack (unbalanced) $A_0 = (\varepsilon, 0, 1 - \varepsilon)$ $A_1 = (\varepsilon/2, \varepsilon/2, 1 - \varepsilon)$	$\varepsilon/2$	$\varepsilon/2$	$\Theta(\varepsilon)$

1.3 Our Results

In this work, in order to evaluate the bit security of hardness amplification directly, we derive a new variant of the XOR lemma in terms of the Rényi advantage. Roughly, our XOR lemma (Theorem 1) claims that if a function is mildly hard in the sense that $\Pr(A(x) = f(x)) \leq 1 - \delta$ for any adversary A of size s, then the Rényi advantage of the XOR function f' satisfies $\mathsf{Adv}^{\mathrm{Renyi}}_{B,G_{f'}} \leq \varepsilon$ for any adversary B of size $s' = \Omega(\varepsilon/\ln(1/\delta)) \cdot s$.[3] This implies that the loss of the reduction is

$$L^{\mathrm{amp}}(n) = \mathcal{O}\left(\frac{\log(1/\delta)}{\varepsilon} \cdot \frac{\varepsilon}{1/2 - \delta}\right) = \mathcal{O}\left(\log(1/\delta)\right),$$

which improves upon the loss in (2) by the factor of $1/\varepsilon$ for general adversaries.

To derive our XOR lemma for the Rényi advantage, we prove a new variant of the hardcore lemma, originally proved by Impagliazzo [23]. Our hardcore lemma is stated in terms of the CS advantage. Then, by using the connection between the CS advantage and the Rényi advantage in [40], we prove the XOR lemma via the hardcore lemma.

To prove our hardcore lemma, we analyze the performance of the boosting algorithm such that weak learners may output the $\perp$ symbol in addition to 0 and 1. Our main technical contribution in this paper is characterizing the performance of the boosting algorithm with $\perp$ in terms of the CS advantage.

Ideas and Techniques. Roughly, our hardcore lemma (Lemma 1) claims that if a function is mildly hard in the sense that $\Pr(A(x) = f(x)) \leq 1 - \delta$ for any adversary A of size s, then there exists a hardcore distribution H with density δ such that the CS advantage of predicting f for inputs distributed according to H satisfies $\mathsf{Adv}^{\mathrm{CS}}_{B,f|H} \leq \varepsilon$ for any adversary B of size $s' = \Omega(\varepsilon/\ln(1/\delta)) \cdot s$. Compared to the standard hardcore lemma known in the literature, which involves the decrease of circuit size by the factor of $\Omega(\tilde{\varepsilon}^2/\ln(1/\delta))$ for the standard advantage $\tilde{\varepsilon}$, the decrease of circuit size in our hardcore lemma is by the factor of $\Omega(\varepsilon/\ln(1/\delta))$ for the CS advantage ε. For balanced adversaries, since the CS advantage behaves as $\varepsilon \simeq \tilde{\varepsilon}^2$, our hardcore lemma is essentially the same as the standard hardcore lemma; however, for unbalanced adversaries, since the CS advantage behaves as $\varepsilon \simeq \tilde{\varepsilon}$, our hardcore lemma improves the decrease of circuit size upon the standard hardcore lemma.

A utility of the CS advantage in the context of the Goldreich-Levin (GL) algorithm has been reported by Hast in [17]; he proposed a modified version of the GL algorithm by taking into account adversaries that may output $\perp$ when predicting the hardcore bit and characterized the performance of such a GL algorithm in terms of the CS advantage. His algorithm was used in [40] to prove

[3] More precisely, this statement assumes that the cost of a weighted majority gate can be ignored. Indeed, if $s = \omega(\log(1/\delta)/\varepsilon^2)$, the cost is negligible; See Remark 1 for discussion.

the tightness of the GL theorem. In this work, we use the utility of outputting $\perp$ in a hardcore lemma to provide a tight reduction of hardness amplification. Since the performance of the GL algorithm can be improved by taking into account the effect of $\perp$, it is natural to seek a hardcore lemma that takes into account the effect of $\perp$ as well. However, such a hardcore lemma has not been proposed in the literature. In fact, it is not immediately clear how $\perp$ can be incorporated into a hardcore lemma. In the case of the GL algorithm, the algorithm can be interpreted as a list decoding of the Hadamard code. Then, it is natural to consider $\perp$ as an erasure of error correcting code. On the other hand, the hardcore lemma is interpreted as the boosting algorithm in the learning theory. Unlike the error correcting code, the role of $\perp$ in the boosting is not so clear. In fact, as we will discuss below, the usage of $\perp$ in our boosting algorithm is subtle, and the effect of $\perp$ only shows up after a judicious analysis of the algorithm.

Readers might wonder why the XOR lemma is stated and proved in terms of the Rényi advantage while the hardcore lemma is stated and proved in terms of the CS advantage. When we prove the standard XOR lemma using the standard hardcore lemma, we decompose the uniform distribution of inputs into a weighted sum of the "easy part" and the "hardcore part." Since the probability of correct prediction is an affine function of the distribution of inputs, it can also be decomposed as the corresponding weighted sum of the probability of correct prediction for the easy and the hardcore parts. A main obstacle to using such a decomposition argument in our setting is that neither the CS advantage nor the Rényi advantage are affine functions of input distributions. Fortunately, by using the joint convexity of the Rényi divergence in a judicious manner, we can go through the decomposition argument for the Rényi advantage; it is not clear if a similar trick can be applied for the CS advantage, which is the reason why we consider the XOR lemma for the Rényi advantage.

On the other hand, while the labeling of the adversary's output symbols is irrelevant to the Rényi advantage, the abort symbol $\perp$ has a special meaning in the CS advantage. Thus, the CS advantage is more suitable to be used when we prove the hardcore lemma by using the boosting with $\perp$.

Heuristically, a gist of the standard boosting algorithm (i.e., weak learners without $\perp$) is to update the input distribution iteratively as follows: for a given weak learner of the current round, we update the distribution of inputs so that the probabilities of symbols correctly predicted by the current weak learner are decreased, and the probabilities of symbols erroneously predicted by the current weak learner are increased; we choose the multiplicative weight (for decrease/increase) of the update so that the advantage of the weak learner of the previous round is zero for the updated distribution. In our setting, weak learners may output $\perp$ in addition to the binary outputs. Then, a natural question is whether we should increase or decrease the probabilities of symbols when the current weak learner output $\perp$. Perhaps surprisingly, our boosting algorithm (Algorithm 1) neither increases nor decreases the probabilities of those symbols,[4]

[4] Note that since there is a renormalization procedure, the updated probabilities of those symbols may be changed.

and an appropriate choice of the multiplicative weight provides the desired strong learner. In fact, we will argue that our update rule is an approximate version of the rule such that the CS advantage of the weak learner of the previous round is zero for the updated distribution (Appendix C).

Future Perspectives. It has been known that the hardcore lemma and its variants play a pivotal role in the construction of cryptographic primitives: for instance, the construction of strong PRGs from weak ones [30] and the constructions of PRGs from one-way functions [16,38]. In order to discuss the bit security of such constructions, we need to take into account unbalanced adversaries, as discussed above. Our hardcore lemma for the CS advantage may have a potential use in that direction of research.

1.4 Related Work

The study of hardness amplification has a long history, and there are several proofs of the XOR lemma; see [10] for a thorough review. As mentioned in Sect. 1.3, in this paper, we prove our XOR lemma along the lines of the proof by Impagliazzo using the hardcore lemma [23].

Another line of studies on hardness amplification is the direct-product constructions; it aims to construct strongly hard (search type) functions from weak ones [11,13,26]. See [26] and the literature therein for recent related work. In this work, we focus on amplifying the hardness of Boolean (decision type) functions.

In the original paper [23], Impagliazzo provided two proofs of the hardcore lemma, a constructive one and one based on the min-max theorem.[5] Later, it was pointed out that the constructive proof can be interpreted as the boosting algorithm in learning theory [25]. Based on such identification, there have been several improvements and applications of the hardcore lemma [3,22,24,30,37].

In contrast to the standard hardcore lemma stated in terms of the probability of correct prediction, our hardcore lemma is stated in terms of the CS advantage. The main difficulty of handling the CS advantage is that it may not be affine with respect to either the input distribution or (stochastic) circuit. Thus, it is unclear if the min-max theorem is applicable to prove the hardcore lemma for the CS advantage. To overcome this difficulty, we devise a modified version of the boosting algorithm in [3,24] by considering that the adversary (weak learner in the context of learning) may output $\perp$ in addition to 0 and 1. In the context of learning theory, by considering the asymmetry of weak learners' confidence for each output, we can improve the standard AdaBoost, which is known as the confidence-rated AdaBoost or the infoBoosting [1,19,20,36]. Our boosting algorithm is closely related in spirit to those algorithms in the sense that the symbol $\perp$ signifies that weak learners' confidence is zero. Since the CS advantage is a criterion initiated in cryptography, we believe it is an interesting contribution to characterize the boosting algorithm with $\perp$ in terms of the CS advantages of weak learners; perhaps, it may have certain applications in learning theory.

[5] The latter was attributed to Nisan.

Exploring a single quantity for measuring the security of primitives dates back to [21,29], where the time-success ratio was proposed. Micciancio and Walter [32] revisited the notion of bit security, especially for decision primitives, and justified the CS advantage. Watanabe and Yasunaga [39] introduced the Rényi advantage to characterize an operational definition of bit security. The two advantages turned out to be (essentially) equivalent when adversaries are allowed to output $\perp$ [40]. Recently, Lee [27] introduced another notion of bit security that captures search and decision games in a single framework.

1.5 Paper Organization

We present the formulation of the hardness amplification and the XOR lemma for the Rényi advantage in Sect. 2. In Sect. 3, we present the hardcore lemma for the CS advantage and its proof using the boosting algorithm with $\perp$. Section 4 presents the proof of the XOR lemma by using the hardcore lemma. Other than the fact of approximation as in (1), we do not use the knowledge of bit security frameworks [32,39]. For readers' convenience, we review the bit-security frameworks in Appendix A.

2 Hardness Amplification for Rényi Advantage

For $0 \le \rho \le 1$ and a function $f : \{0,1\}^n \to \{0,1\}$, ρ-hardness $\mathrm{H}^{\rho}_{\mathsf{avg}}(f)$ of function f is the largest integer s such that any circuit $C : \{0,1\}^n \to \{0,1,\perp\}$ of size at most s satisfies

$$\Pr_{x \sim U_n}\big(C(x) = f(x)\big) \le \rho,$$

where U_n is the uniform distribution on $\{0,1\}^n$. For a prescribed (typically small) margin $\delta > 0$, a function f is regarded as *mildly hard* if the value of $\mathrm{H}^{1-\delta}_{\mathsf{avg}}(f)$ is sufficiently large. By using the function f as a building block, we are interested in constructing another function that is much harder than f itself. A typically used construction is the so-called XOR construction: for a given integer $k \ge 2$, let $f^{\oplus k} : \{0,1\}^{nk} \to \{0,1\}$ be the function defined by

$$f^{\oplus k}(x_1, \ldots, x_k) := f(x_1) \oplus \cdots \oplus f(x_k),$$

where $x_1, \ldots, x_k \in \{0,1\}^n$. The standard XOR lemma of Yao claims that $f^{\oplus k}$ is hard in the sense that $\mathrm{H}^{1/2+\varepsilon}_{\mathsf{avg}}(f^{\oplus k})$ is as large as $\frac{\varepsilon^2}{\ln(1/\delta)}\mathrm{H}^{1-\delta}_{\mathsf{avg}}(f)$ for $\varepsilon \ge 2(1-\delta)^k$; this means that even though the circuit size is decreased by the factor of $\frac{\varepsilon^2}{\ln(1/\delta)}$, we can guarantee that the adversary's success probability of predicting the value of function $f^{\oplus k}$ is at most $\frac{1}{2} + \varepsilon$. More precisely, the following holds:

Proposition 1 (XOR lemma). *For $\varepsilon \ge 2(1-\delta)^k$, it holds that*

$$\Pr_{x_1,\ldots,x_k \sim U_n}\big(C(x_1, \ldots, x_k) = f^{\oplus k}(x_1, \ldots, x_k)\big) \le \frac{1}{2} + \varepsilon$$

for every circuit C of size at most s for $s = \Omega\left(\frac{\varepsilon^2}{\ln(1/\delta)}\right) \cdot \mathrm{H}^{1-\delta}_{\mathsf{avg}}(f)$.

In order to discuss the bit security of the XOR function, let us consider the distinguishing game between $u = 0$ instance $(x_1, \ldots, x_k, f^{\oplus k}(x_1, \ldots, x_k))$ and $u = 1$ instance $(x_1, \ldots, x_k, \sigma)$, where σ is a random bit that is independent of $(x_1, \ldots, x_k)$. Proposition 1 implies that (by the standard argument of converting a distinguisher to a predictor), for every circuit of size at most s, the standard distinguishing advantage (in terms of the total variation distance) is less than ε. However, as discussed in [32,39] (see also [40] for more detail), the standard advantage is not suitable for evaluating bit security. Thus, the above-mentioned XOR lemma does not guarantee that the bit security is preserved during the process of constructing $f^{\oplus k}$ from f. To resolve this issue, we derive an alternative version of the XOR lemma in terms of Rényi advantage

$$\mathsf{Adv}^{\mathrm{Renyi}}_{A, f^{\oplus k}} = D_{1/2}(A_0 \| A_1) = -2 \ln \sum_a \sqrt{A_0(a) A_1(a)},$$

where A_u is the distribution of the output by adversary when the instance is u. As we mentioned in Sect. 1 (see also Appendix A), the bit security can be approximated by the Rényi advantage up to a constant. To that end, it is desirable to derive a trade-off between the adversary's Rényi advantage and the circuit size. We use the weighted majority gate once in the reduction proof of the following theorem. To avoid the effect of how the weighted majority gate is implemented, we first assume that the weighted majority gate is available for free in the evaluation of the initial hardness $\mathrm{H}^{1-\delta}_{\mathsf{avg}}(f)$; in Remark 1, we will provide an estimate for the effect of implementing the weighted majority gate.

Theorem 1 (XOR lemma for Rényi advantage). *For $\varepsilon \geq 2(1-\delta)^k$, it holds that*

$$\mathsf{Adv}^{\mathrm{Renyi}}_{A, f^{\oplus k}} \leq \varepsilon$$

for every circuit A of size $s' \leq \frac{\varepsilon}{48 \ln(1/\delta)} \mathrm{H}^{1-\delta}_{\mathsf{avg}}(f)$, where the initial hardness $\mathrm{H}^{1-\delta}_{\mathsf{avg}}(f)$ is evaluated under the assumption that the weighted majority gate is available for free.

Remark 1. The assumption of the availability of the weighted majority gate comes from the fact that it is used in the proof of the hardcore lemma of Lemma 1. As we discuss in Remark 2, the effect of implementing the weighted majority gate can be estimated. More specifically, the statement of Theorem 1 holds for circuit size $s' \leq \frac{\varepsilon}{64 \ln(1/\delta)} \mathrm{H}^{1-\delta}_{\mathsf{avg}}(f) - \frac{c}{\varepsilon}$ for some constant c. Thus, when the initial hardness is $\mathrm{H}^{1-\delta}_{\mathsf{avg}}(f) = \omega(\log(1/\delta)/\varepsilon^2)$, then the effect of the weighted majority is negligible.

We shall discuss an implication of Theorem 1. For a given integer s, the bit security against adversaries with cost s is evaluated as (1). In fact, in the setting of this section, the initial hardness $s = \mathrm{H}^{1-\delta}_{\mathsf{avg}}(f)$ means $\mathrm{BS}_s(G_f) = \log s + \mathcal{O}(1)$. For the function f itself, since circuits of much smaller size s' may have the same

success probability $1-\delta$, we cannot guarantee $\mathrm{BS}_{s'}(G_f) \geq \mathrm{BS}_s(G_f)$. However, Theorem 1 implies that the XOR function $f^{\oplus k}$ satisfies

$$\mathrm{BS}_{s'}(G_{f^{\oplus k}}) \geq \mathrm{BS}_s(G_f) - \mathcal{O}(\log \ln(1/\delta))$$

for $s' = \frac{\varepsilon}{48 \ln(1/\delta)} \mathrm{H}^{1-\delta}_{\mathsf{avg}}(f)$. In this sense, the bit security is preserved in the hardness amplification.

3 Hardcore Lemma for CS Advantage

We shall prove Theorem 1 along the lines of the proof by Impagliazzo using the hardcore lemma [23]. To that end, we develop a new variant of the hardcore lemma in this section.

By the definition of hardness, any circuit C of size $s \leq \mathrm{H}^{1-\delta}_{\mathsf{avg}}(f)$ must satisfy

$$\Pr_{x \sim U_n}\big(C(x) = f(x)\big) \leq 1 - \delta. \tag{3}$$

This means that there exists a set $\mathcal{H}_C \subset \{0,1\}^n$ of hard inputs such that $|\mathcal{H}_C| \geq \delta 2^n$ and the circuit C fails to compute $f(x)$ for every $x \in \mathcal{H}_C$; however, the hard sets may differ for different circuits. Impagliazzo's hardcore lemma claims that there exists a set of inputs that are universally hard for every circuit with a smaller size. It is more convenient to consider probability distributions, rather than sets, having density δ; a distribution P on $\{0,1\}^n$ is said to have density δ if $P(x) \leq \frac{1}{\delta 2^n}$ for every $x \in \{0,1\}^n$, or equivalently, the min-entropy satisfies $H_{\min}(P) \geq n - \log(1/\delta)$. The standard hardcore lemma is a statement as follows:

Proposition 2 (Hardcore lemma). *There exists a hardcore distribution H having density δ such that*

$$\Pr_{x \sim H}\big(C(x) = f(x)\big) \leq \frac{1}{2} + \varepsilon \tag{4}$$

for every circuit C of size at most s for $s = \Omega\left(\frac{\varepsilon^2}{\ln(1/\delta)}\right) \cdot \mathrm{H}^{1-\delta}_{\mathsf{avg}}(f)$.

Since the standard hardcore lemma, Lemma 2, is insufficient to prove Theorem 1, we derive the following variant of the hardcore lemma in terms of the conditional squared (CS) advantage. For a given distribution P on $\{0,1\}^n$ and

a circuit $C : \{0,1\}^n \to \{0,1,\bot\}$, the CS advantage of predicting f is defined as[6]

$$\mathsf{Adv}^{\mathrm{CS}}_{C,f|P} := \frac{4\left(\Pr(C(x) = f(x)) - \frac{1}{2}\Pr(C(x) \neq \bot)\right)^2}{\Pr(C(x) \neq \bot)} \tag{5}$$

$$= \frac{\left(\Pr(C(x) = f(x)) - \Pr(C(x) = \overline{f(x)})\right)^2}{\Pr(C(x) \neq \bot)} \tag{6}$$

where the probability is with respect to $x \sim P$ and $\overline{f(x)} = f(x) \oplus 1$.

Lemma 1 (Hardcore lemma for CS advantage). *There exists a hardcore distribution H having density δ such that*

$$\mathsf{Adv}^{\mathrm{CS}}_{C,f|H} \leq \varepsilon \tag{7}$$

for every circuit C of size at most $s' := \frac{\varepsilon}{8\ln(1/\delta)}\mathrm{H}^{1-\delta}_{\mathsf{avg}}(f)$.

For a circuit that does not output $\bot$, i.e., $\Pr(C(x) \neq \bot) = 1$, we can rewrite (5) as

$$\Pr(C(x) = f(x)) = \frac{1}{2} + \frac{\sqrt{\mathsf{Adv}^{\mathrm{CS}}_{C,f|P}}}{2}.$$

For such a circuit, the bounds (4) and (7) are the same up to a constant (ε^2 in Proposition 2 corresponds to ε in Lemma 1). A main new feature of Lemma 1 is that it can be applied to circuits that may output $\bot$ with significant probability.

In contrast to the standard correct probability (the left-hand side of (4)), the CS advantage is not affine with respect to either the input distribution or (stochastic) circuit. Thus, it is unclear if the min-max theorem is applicable to prove Lemma 1. Instead, we consider a modified version of the boosting algorithm by taking into account the fact that the adversary (weak learner in the context of learning) may output $\bot$ in addition to 0 and 1.

To prove Lemma 1 via a contradiction, suppose that for each distribution P having density δ, there exists a circuit C_P of size at most s' such that

$$\mathsf{Adv}^{\mathrm{CS}}_{C_P,f|P} > \varepsilon. \tag{8}$$

Starting from the uniform distribution $P^{(1)}$ and a circuit that satisfies (8) for $P^{(1)}$, we are going to sequentially update distributions and corresponding circuits that satisfy (8); then, by combining those circuits, we eventually construct a circuit that violates the assumption (3) on the hardness of f. As we mentioned

[6] More precisely, the CS advantage in (5) is for a predictor; on the other hand, when we define the bit security, we consider the CS advantage for a distinguisher (cf. (27)). The CS advantage for a predictor was first introduced in the context of the Goldreich-Levin algorithm [17].

Algorithm 1: Boosting

Input: The number $T \in \mathbb{N}$ of iteration and a circuit C_P satisfying (8) for each P with density δ

Output: A sequence of circuits $C_{P^{(1)}}, \ldots, C_{P^{(T)}}$

1: Initialize $P^{(1)}$ as the uniform distribution on $\{0,1\}^n$;

2: Repeat Step 3 and Step 4 for $1 \le t \le T$;

3: For a circuit $C_{P^{(t)}}$ that satisfies (8) for $P^{(t)}$, set $\gamma_t = \frac{\Delta_t}{4\alpha_t}$ for

$$\alpha_t := \Pr_{x \sim P^{(t)}}\left(C_{P^{(t)}}(x) \neq \perp\right),$$
$$\Delta_t := \Pr_{x \sim P^{(t)}}\left(C_{P^{(t)}}(x) = f(x)\right) - \Pr_{x \sim P^{(t)}}\left(C_{P^{(t)}}(x) = \overline{f(x)}\right),$$

and set

$$\hat{P}^{(t+1)}(x) = \frac{P^{(t)}(x)\exp\left(-\gamma_t\left\{\mathbf{1}[C_{P^{(t)}}(x) = f(x)] - \mathbf{1}[C_{P^{(t)}}(x) = \overline{f(x)}]\right\}\right)}{Z_{P^{(t)}}},$$

where $\overline{f(x)} = f(x) \oplus 1$ and

$$Z_{P^{(t)}} = \sum_x P^{(t)}(x)\exp\left(-\gamma_t\left\{\mathbf{1}[C_{P^{(t)}}(x) = f(x)] - \mathbf{1}[C_{P^{(t)}}(x) = \overline{f(x)}]\right\}\right)$$

is the normalizer;

4: For the set $\mathcal{P}_\delta$ of all distributions with density δ, set

$$P^{(t+1)} = \operatorname*{argmin}_{P \in \mathcal{P}_\delta} D(P \| \hat{P}^{(t+1)}),$$

where D is the KL-divergence.

above, this procedure is essentially the same as the boosting algorithm in learning theory, in which we construct a strong learner from weak learners. Our algorithm for boosting is described in Algorithm 1. Perhaps surprisingly, the update rule of Algorithm 1 does not alter the distribution when the output of a circuit is $\perp$; a rationale for the algorithm will be discussed in Appendix C.

The sequence of distributions generated by the algorithm satisfies the following.

Lemma 2. *The distributions $P^{(1)}, \ldots, P^{(T)}$ generated by Algorithm 1 satisfy*

$$\sum_{t=1}^{T} \frac{1}{T} \frac{\mathsf{Adv}^{\mathrm{CS}}_{C_P^{(t)}, f|P^{(t)}}}{8} \le \mathbb{E}_{x \sim P}\left[\sum_{t=1}^{T} \frac{1}{T}\gamma_t\left\{\mathbf{1}[C_{P^{(t)}}(x) = f(x)] - \mathbf{1}[C_{P^{(t)}}(x) = \overline{f(x)}]\right\}\right] + \frac{D(P\|P^{(1)})}{T} \tag{9}$$

for every $P \in \mathcal{P}_\delta$, where $\mathcal{P}_\delta$ is the set of all distributions with density δ.

Lemma 2 is a counterpart of technical results that appeared in the literature for the boosting without $\perp$ (e.g. [24, Theorem 1] or [3, Lemma 4.1]). Since our boosting algorithm invokes circuits that may output $\perp$, the performance is evaluated in terms of the CS advantage instead of the standard advantage. A main technical difficulty of proving Lemma 2 is connecting Algorithm 1 to the CS advantage since the output $\perp$ does not appear explicitly in the algorithm. The following lemma is a crucial step in proving Lemma 2, which enables us to evaluate the performance of Algorithm 1 in terms of the CS advantage. The lemma will be proved in Sect. 3.1.

Lemma 3. *For every $1 \le t \le T$, the distributions $P^{(t)}$ and $\hat{P}^{(t+1)}$ in Algorithm 1 satisfy*

$$D(P^{(t)} \| \hat{P}^{(t+1)}) \le 2\gamma_t^2 \Pr_{x \sim P^{(t)}} \left(C_{P^{(t)}}(x) \neq \perp \right). \tag{10}$$

Since we set $\gamma_t = \frac{\Delta_t}{4\alpha_t}$ in Algorithm 1, note that the right-hand side of (10) is $\frac{\Delta_t^2}{8\alpha_t} = \frac{1}{8}\mathsf{Adv}^{\mathrm{CS}}_{C_P^{(t)}, f|P^{(t)}}$. In this manner, the CS advantage shows up in the analysis even though the output $\perp$ does not appear in the algorithm explicitly. The details of the proof of Lemma 2 are presented in Sect. 3.2.

Once we prove Lemma 2, we can prove the hardcore lemma for the CS advantage (Lemma 1) via a contrapositive argument: we assume that for each distribution P having density δ, there exists a circuit C_P of size at most s' satisfying (8). This assumption enables us to find a weak learner in each iteration of Algorithm 1, and we can construct a strong learner from a sequence of circuits found by the algorithm. One key difference from the proof of the standard hardcore lemma is that, instead of the standard majority, we take the weighted majority to construct the strong learner from weak learners since reliabilities of weak learners vary depending on the probability of output $\perp$. The detailed proof of Lemma 1 is provided in Sect. 3.3.

3.1 Proof of Lemma 3

From the update rule of Algorithm 1, we have

$$\begin{aligned}
&D(P^{(t)} \| \hat{P}^{(t+1)}) \\
&= \sum_x P^{(t)}(x)\gamma_t \left\{ \mathbf{1}[C_{P^{(t)}}(x) = f(x)] - \mathbf{1}[C_{P^{(t)}}(x) = \overline{f(x)}] \right\} + \ln Z_{P^{(t)}} \\
&= \gamma_t \left\{ \Pr_{x \sim P^{(t)}} \left(C_{P^{(t)}}(x) = f(x) \right) - \Pr_{x \sim P^{(t)}} \left(C_{P^{(t)}}(x) = \overline{f(x)} \right) \right\} + \ln Z_{P^{(t)}}. \qquad (11)
\end{aligned}$$

Now, we evaluate $\ln Z_{P^{(t)}}$ as

$$\begin{aligned}
\ln Z_{P^{(t)}} &= \ln \sum_x P^{(t)}(x) \exp\big(-\gamma_t\{\mathbf{1}[C_{P^{(t)}}(x) = f(x)] - \mathbf{1}[C_{P^{(t)}}(x) = \overline{f(x)}]\}\big) \\
&\le \ln \sum_x P^{(t)}(x)\big(1 - \gamma_t\{\mathbf{1}[C_{P^{(t)}}(x) = f(x)] - \mathbf{1}[C_{P^{(t)}}(x) = \overline{f(x)}]\} \\
&\qquad + 2\gamma_t^2 \mathbf{1}[C_{P^{(t)}}(x) \neq \perp]\big) \\
&= \ln \Bigg(1 - \gamma_t\Big\{ \Pr_{x\sim P^{(t)}}\big(C_{P^{(t)}}(x) = f(x)\big) - \Pr_{x\sim P^{(t)}}\big(C_{P^{(t)}}(x) = \overline{f(x)}\big)\Big\} \\
&\qquad + 2\gamma_t^2 \Pr_{x\sim P^{(t)}}\big(C_{P^{(t)}}(x) \neq \perp\big)\Bigg) \\
&\le -\gamma_t\Big\{ \Pr_{x\sim P^{(t)}}\big(C_{P^{(t)}}(x) = f(x)\big) - \Pr_{x\sim P^{(t)}}\big(C_{P^{(t)}}(x) = \overline{f(x)}\big)\Big\} \\
&\qquad + 2\gamma_t^2 \Pr_{x\sim P^{(t)}}\big(C_{P^{(t)}}(x) \neq \perp\big),
\end{aligned} \tag{12}$$

where the first inequality follows from[7] $e^{-\theta} \le 1 - \theta + 2\theta^2$ for $\theta \in [-1, 1]$ and that

$$\exp\big(-\gamma_t\{\mathbf{1}[C_{P^{(t)}}(x) = f(x)] - \mathbf{1}[C_{P^{(t)}}(x) = \overline{f(x)}]\}\big) = 1$$

when $C_{P^{(t)}}(x) = \perp$; and the second inequality follows from $\ln(1-\theta) \le -\theta$ for $\theta < 1$. By combining (11) and (12), we have the desired bound. □

3.2 Proof of Lemma 2

By the cosine law of the KL-divergence, we have

[7] By the Taylor approximation, we have

$$e^{-\theta} \le 1 - \theta + \sup_{-1\le\tau\le 1} \frac{e^{-\tau}}{2}\theta^2 \le 1 - \theta + \frac{e}{2}\theta^2 \le 1 - \theta + 2\theta^2$$

.

$$
\begin{aligned}
&D(P\|P^{(t)}) - D(P\|\hat{P}^{(t+1)}) \\
&= \sum_x \big(P^{(t)}(x) - P(x)\big)\big(\ln P^{(t)}(x) - \ln \hat{P}^{(t+1)}(x)\big) - D(P^{(t)}\|\hat{P}^{(t+1)}) \\
&= \sum_x \big(P^{(t)}(x) - P(x)\big)\gamma_t\big\{\mathbf{1}[C_{P^{(t)}}(x) = f(x)] - \mathbf{1}[C_{P^{(t)}}(x) = \overline{f(x)}]\big\} \\
&\qquad\qquad - D(P^{(t)}\|\hat{P}^{(t+1)}) \\
&= \gamma_t\big\{ \Pr_{x\sim P^{(t)}}\big(C_{P^{(t)}}(x) = f(x)\big) - \Pr_{x\sim P^{(t)}}\big(C_{P^{(t)}}(x) = \overline{f(x)}\big)\big\} \\
&\quad - \gamma_t \mathbb{E}_{x\sim P}\Big[\big\{\mathbf{1}[C_{P^{(t)}}(x) = f(x)] - \mathbf{1}[C_{P^{(t)}}(x) = \overline{f(x)}]\big\}\Big] - D(P^{(t)}\|\hat{P}^{(t+1)}) \\
&\geq \gamma_t\big\{ \Pr_{x\sim P^{(t)}}\big(C_{P^{(t)}}(x) = f(x)\big) - \Pr_{x\sim P^{(t)}}\big(C_{P^{(t)}}(x) = \overline{f(x)}\big)\big\} \\
&\quad - \gamma_t \mathbb{E}_{x\sim P}\Big[\big\{\mathbf{1}[C_{P^{(t)}}(x) = f(x)] - \mathbf{1}[C_{P^{(t)}}(x) = \overline{f(x)}]\big\}\Big] \\
&\qquad\qquad - 2\gamma_t^2 \Pr_{x\sim P^{(t)}}\big(C_{P^{(t)}}(x) \neq \perp\big), \qquad (13)
\end{aligned}
$$

where, in the second equality, we used the fact that $\ln Z_{P^{(t)}}$ does not depend on x and $\sum_x \big(P^{(t)}(x) - P(x)\big) = 0$ to eliminate $\ln Z_{P^{(t)}}$; and we used Lemma 3 in the final inequality. Here, we apply the Pythagorean inequality by noting that $\mathcal{P}_\delta$ is a closed convex set: it holds that (e.g., see [5, Theorem 3.1])

$$
D(P\|P^{(t+1)}) + D(P^{(t+1)}\|\hat{P}^{(t+1)}) \leq D(P\|\hat{P}^{(t+1)})
$$

for any $P \in \mathcal{P}_\delta$, which implies

$$
D(P\|P^{(t+1)}) \leq D(P\|\hat{P}^{(t+1)}). \qquad (14)
$$

Thus, (13) and (14) imply

$$
\begin{aligned}
&D(P\|P^{(t)}) - D(P\|P^{(t+1)}) \\
&\geq \gamma_t\big\{ \Pr_{x\sim P^{(t)}}\big(C_{P^{(t)}}(x) = f(x)\big) - \Pr_{x\sim P^{(t)}}\big(C_{P^{(t)}}(x) = \overline{f(x)}\big)\big\} \\
&\quad - \gamma_t \mathbb{E}_{x\sim P}\Big[\big\{\mathbf{1}[C_{P^{(t)}}(x) = f(x)] - \mathbf{1}[C_{P^{(t)}}(x) = \overline{f(x)}]\big\}\Big] \\
&\quad - 2\gamma_t^2 \Pr_{x\sim P^{(t)}}\big(C_{P^{(t)}}(x) \neq \perp\big).
\end{aligned}
$$

By taking the summation of both sides for $t = 1$ through T, we have

$$
\begin{aligned}
&D(P\|P^{(1)}) - D(P\|P^{(T+1)}) \\
&\geq \sum_{t=1}^{T} \gamma_t\big\{ \Pr_{x\sim P^{(t)}}\big(C_{P^{(t)}}(x) = f(x)\big) - \Pr_{x\sim P^{(t)}}\big(C_{P^{(t)}}(x) = \overline{f(x)}\big)\big\} \\
&\quad - \mathbb{E}_{x\sim P}\Big[\sum_{t=1}^{T} \gamma_t\big\{\mathbf{1}[C_{P^{(t)}}(x) = f(x)] - \mathbf{1}[C_{P^{(t)}}(x) = \overline{f(x)}]\big\}\Big] \\
&\quad - \sum_{t=1}^{T} 2\gamma_t^2 \Pr_{x\sim P^{(t)}}\big(C_{P^{(t)}}(x) \neq \perp\big). \qquad (15)
\end{aligned}
$$

Since $D(P\|P^{(T+1)}) \geq 0$, by substituting α_t, Δ_t and $\gamma_t = \frac{\Delta_t}{4\alpha_t}$ defined in Algorithm 1, and by rearranging terms, we have

$$\sum_{t=1}^{T} \frac{\Delta_t^2}{8\alpha_t}$$
$$\leq \mathbb{E}_{x\sim P}\left[\sum_{t=1}^{T} \gamma_t\left\{\mathbf{1}[C_{P^{(t)}}(x) = f(x)] - \mathbf{1}[C_{P^{(t)}}(x) = \overline{f(x)}]\right\}\right] + D(P\|P^{(1)}).$$

Finally, by noting that $\mathsf{Adv}^{\mathrm{CS}}_{C_P^{(t)}, f|P^{(t)}} = \frac{\Delta_t^2}{\alpha_t}$ and by dividing by T, we have (9). □

3.3 Proof of Lemma 1

To prove via a contradiction, suppose that for each distribution P having density δ, there exists a circuit C_P of size at most s' satisfying (8). We shall prove that there exists a circuit $C^\star$ of size at most $s := \mathrm{H}^{1-\delta}_{\mathsf{avg}}(f)$ such that

$$\Pr_{x\sim U_n}\left(C^\star(x) = f(x)\right) > 1 - \delta. \tag{16}$$

We construct $C^\star$ as follows. For $T = \left\lceil \frac{8\ln(1/\delta)}{\varepsilon} \right\rceil$, let $C_{P^{(1)}}, \ldots, C_{P^{(T)}}$ be the circuits obtained by Algorithm 1. For a given input $x \in \{0,1\}^n$, by invoking the weighted majority oracle, $C^\star$ outputs $a \in \{0,1\}$ if (the tie can be decided arbitrarily)

$$\sum_{t=1}^{T} \frac{1}{T}\gamma_t\left\{\mathbf{1}[C_{P^{(t)}}(x) = a] - \mathbf{1}[C_{P^{(t)}}(x) = \overline{a}]\right\} > 0, \tag{17}$$

where $\overline{a} = a \oplus 1$. Note that the size of $C^\star$ is Ts'. Note also that $C^\star$ makes an error for input x, i.e., $C^\star(x) = \overline{f(x)}$ only if

$$\sum_{t=1}^{T} \frac{1}{T}\gamma_t\left\{\mathbf{1}[C_{P^{(t)}}(x) = f(x)] - \mathbf{1}[C_{P^{(t)}}(x) = \overline{f(x)}]\right\} \leq 0. \tag{18}$$

Let

$$\mathcal{E} = \left\{x \in \{0,1\}^n : \sum_{t=1}^{T} \frac{1}{T}\gamma_t\left\{\mathbf{1}[C_{P^{(t)}}(x) = f(x)] - \mathbf{1}[C_{P^{(t)}}(x) = \overline{f(x)}]\right\} \leq 0\right\}.$$

be the set of all inputs such that $C^\star$ may make an error. If we prove $\frac{|\mathcal{E}|}{2^n} < \delta$, we are done, i.e., (16) holds. To prove via a contradiction, assume that $\frac{|\mathcal{E}|}{2^n} \geq \delta$, which implies that the uniform distribution $P_{\mathcal{E}}$ on $\mathcal{E}$ has density δ. Since $P^{(1)}$ is the uniform distribution, we have

$$D(P_{\mathcal{E}}\|P^{(1)}) = \sum_x P_{\mathcal{E}}(x)\ln 2^n P_{\mathcal{E}}(x) \leq \sum_x P_{\mathcal{E}}(x)\ln(1/\delta) = \ln(1/\delta).$$

By applying Lemma 2 for $P_{\mathcal{E}}$, by noting (8) for each $C_{P^{(t)}}$, and by noting that (18) with probability 1 for $x \sim P_{\mathcal{E}}$, we have

$$\begin{aligned}\frac{\varepsilon}{8} &< \mathbb{E}_{x\sim P_{\mathcal{E}}}\left[\sum_{t=1}^{T}\frac{1}{T}\gamma_t\left\{\mathbf{1}[C_{P^{(t)}}(x)=f(x)]-\mathbf{1}[C_{P^{(t)}}(x)=\overline{f(x)}]\right\}\right]+\frac{D(P_{\mathcal{E}}\|P^{(1)})}{T}\\ &\le \frac{D(P_{\mathcal{E}}\|P^{(1)})}{T}\\ &\le \frac{\varepsilon}{8},\end{aligned}$$

which is a contradiction. □

Remark 2. Even though we did not take into account the precision of the weight γ_t in the proof of Lemma 1, it can be evaluated as follows. Note that α_t and Δ_t in Algorithm 1 satisfy

$$\varepsilon < \frac{\Delta_t^2}{\alpha_t} \le \frac{\Delta_t}{\alpha_t} \le 1.$$

By setting $\tau = \lceil \log(1/\varepsilon) \rceil$ so that $\frac{1}{2^\tau} \le \varepsilon$, we divide the interval $(1/2^\tau, 1]$ into τ parts

$$\left(\frac{1}{2^\tau}, \frac{1}{2^{\tau-1}}\right], \left(\frac{1}{2^{\tau-1}}, \frac{1}{2^{\tau-2}}\right], \ldots, \left(\frac{1}{2}, 1\right].$$

Then, each $\frac{\Delta_t}{\alpha_t}$ satisfies

$$\frac{\Delta_t}{\alpha_t} \in \left(\frac{1}{2^{\ell_t}}, \frac{1}{2^{\ell_t-1}}\right]$$

for some ℓ_t; if we set $\gamma_t = \frac{1}{2^{\ell_t+2}}$, then we have

$$\frac{\Delta_t}{8\alpha_t} \le \gamma_t < \frac{\Delta_t}{4\alpha_t}. \tag{19}$$

If we use $\gamma_t = \frac{1}{2^{\ell_t+2}}$ instead of $\gamma_t = \frac{\Delta_t}{4\alpha_t}$ in Algorithm 1, the last two terms of (15) is lower bounded as

$$\begin{aligned}&\sum_{t=1}^{T}\gamma_t\Big\{\Pr_{x\sim P^{(t)}}\left(C_{P^{(t)}}(x)=f(x)\right)-\Pr_{x\sim P^{(t)}}\left(C_{P^{(t)}}(x)=\overline{f(x)}\right)\Big\}\\ &\quad-\sum_{t=1}^{T}2\gamma_t^2\Pr_{x\sim P^{(t)}}\left(C_{P^{(t)}}(x)\neq\perp\right)\\ &=\sum_{t=1}^{T}\frac{\Delta_t^2}{\alpha_t}\gamma_t\frac{\alpha_t}{\Delta_t}\left(1-2\gamma_t\frac{\alpha_t}{\Delta_t}\right)\\ &\ge\sum_{t=1}^{T}\frac{\Delta_t^2}{\alpha_t}\frac{1}{8}\left(1-\frac{2}{8}\right)=\sum_{t=1}^{T}\frac{3\Delta_t^2}{32\alpha_t},\end{aligned}$$

where the inequality follows from that the function $g(\theta) = \theta(1-2\theta)$ is lower bounded by $g(1/8)$ for $1/8 \le \theta \le 1/4$ and (19). Thus, even if we take into account the precision of the weight γ_t, we have the same claim as Lemma 1 except that the factor $\frac{1}{8}$ in s' is replaced by $\frac{3}{32}$.

Furthermore, since each weight γ_t takes a value between $1/2^{\tau+2}$ and $1/8$, we can implement the weighted majority by creating $2^{\tau+2}\gamma_t \le 1/\varepsilon$ copies of each input and by using the majority.[8] If we take into account the cost of the weighted majority, the proof goes through as long as

$$Ts' + \frac{cT}{\varepsilon} = T\left(s' + \frac{c}{\varepsilon}\right) \le s$$

for some constant c.[9] Thus, the claim of Lemma 1 still holds for every circuit of size at most $s' = \frac{3\varepsilon}{32\ln(1/\delta)}\mathrm{H}^{1-\delta}_{\mathsf{avg}}(f) - \frac{c}{\varepsilon}$; when the initial hardness is $\mathrm{H}^{1-\delta}_{\mathsf{avg}}(f) = \omega(\log(1/\delta)/\varepsilon^2)$, then the cost of the weighted majority is negligible.

4 Proof of Theorem 1

For notational simplicity, we prove the case of $k = 2$; general $k \ge 2$ can be proved similarly. Toward deriving a contradiction, suppose that there exists A with size $s' \le \frac{\varepsilon}{48\ln(1/\delta)}\mathrm{H}^{1-\delta}_{\mathsf{avg}}(f)$ such that $\mathsf{Adv}^{\mathrm{Renyi}}_{A,f^{\oplus k}} > \varepsilon$. By Lemma 1, there exists a hardcore distribution H with density δ such that

$$\mathsf{Adv}^{\mathrm{CS}}_{C,f|H} \le \frac{\varepsilon}{6} \tag{20}$$

for every circuit C of size $s' \le \frac{\varepsilon}{48\ln(1/\delta)}\mathrm{H}^{1-\delta}_{\mathsf{avg}}(f)$. Let G be the distribution on $\{0,1\}^n$ given by $G(x) = \frac{1/2^n - \delta H(x)}{1-\delta}$. Then, the uniform distribution U_n on $\{0,1\}^n$ can be decomposed as

$$U_n(x) = (1-\delta)G(x) + \delta H(x). \tag{21}$$

When (x_1, x_2) are distributed according to distribution Q, we denote the output distribution of adversary A by A^Q_u for $u = 0, 1$ (note that, when $u = 1$, σ is generated independently of $(x_1, x_2) \sim Q$). Then, the output distribution A_u of adversary A under the uniform distribution can be decomposed as

$$A_u = (1-\delta)^2 A^{GG}_u + (1-\delta)\delta A^{GH}_u + \delta(1-\delta)A^{HG}_u + \delta^2 A^{HH}_u. \tag{22}$$

[8] Such a naive implementation of the weighted majority has been studied in the circuit complexity [9].

[9] It comes from the fact that the majority can be realized by a circuit of linear size of inputs [41].

By the joint convexity of the Rényi divergence of order 1/2 (e.g., see [8, Theorem 11]), we have

$$\begin{aligned}\varepsilon &< \mathsf{Adv}^{\mathrm{Renyi}}_{A,f^{\oplus k}} \\ &= D_{1/2}(A_0 \| A_1) \\ &\le (1-\delta)^2 D_{1/2}(A_0^{GG} \| A_1^{GG}) + (1-\delta)\delta D_{1/2}(A_0^{GH} \| A_1^{GH}) \\ &\quad + \delta(1-\delta) D_{1/2}(A_0^{HG} \| A_1^{HG}) + \delta^2 D_{1/2}(A_0^{HH} \| A_1^{HH}).\end{aligned}$$

Since $(1-\delta)^2 < \frac{\varepsilon}{2}$ by the assumption and $D_{1/2}(A_0^{GG} \| A_1^{GG}) \le 1$ (cf. [40, Proposition 1]), we have

$$\begin{aligned}\frac{\varepsilon}{2} &< (1-\delta)\delta D_{1/2}(A_0^{GH} \| A_1^{GH}) \\ &\quad + \delta(1-\delta) D_{1/2}(A_0^{HG} \| A_1^{HG}) + \delta^2 D_{1/2}(A_0^{HH} \| A_1^{HH}).\end{aligned}$$

Since the summation of $(1-\delta)\delta$, $\delta(1-\delta)$, and δ^2 is less than 1, by the averaging argument, at least one of $D_{1/2}(A_0^{GH} \| A_1^{GH})$, $D_{1/2}(A_0^{HG} \| A_1^{HG})$, or $D_{1/2}(A_0^{HH} \| A_1^{HH})$ is larger than $\frac{\varepsilon}{2}$. For instance, suppose that $D_{1/2}(A_0^{GH} \| A_1^{GH}) > \frac{\varepsilon}{2}$ (other cases are similar). We can write

$$\begin{aligned}A_0^{GH}(a) &= \Pr_{\substack{x\sim G \\ y\sim H}}\left(A(x,y,f(x)\oplus f(y)) = a\right) \\ &= \mathbb{E}_{x\sim G}\left[\Pr_{y\sim H}\left(A(x,y,f(x)\oplus f(y)) = a\right)\right]\end{aligned}$$

and

$$\begin{aligned}A_1^{GH}(a) &= \Pr_{\substack{x\sim G \\ y\sim H}}\left(A(x,y,\sigma) = a\right) \\ &= \Pr_{\substack{x\sim G \\ y\sim H}}\left(A(x,y,f(x)\oplus \sigma) = a\right) \qquad (23) \\ &= \mathbb{E}_{x\sim G}\left[\Pr_{y\sim H}\left(A(x,y,f(x)\oplus \sigma) = a\right)\right],\end{aligned}$$

where the identity (23) holds since σ being a random bit independent of (x,y) implies that $f(x)\oplus\sigma$ is a random bit independent of (x,y). For each $x \in \{0,1\}^n$, let us consider an adversary A^x for distinguishing between $(y, f(y))$ and (y,σ) for $y \sim H$; given input (y,z), A^x runs $A(x,y,f(x)\oplus z)$ (since we consider non-uniform complexity, $f(x)$ can be precomputed and provided to the circuit, and the size of A^x is s'). By applying the joint convexity of the Rényi divergence of order 1/2 once more, we have

$$\begin{aligned}\frac{\varepsilon}{2} &< D_{1/2}(A_0^{GH} \| A_1^{GH}) \qquad (24) \\ &\le \mathbb{E}_{x\sim G}\left[D_{1/2}(A_0^x \| A_1^x)\right]. \qquad (25)\end{aligned}$$

Thus, there exists $x \in \{0,1\}^n$ such that

$$\frac{\varepsilon}{2} < D_{1/2}(A_0^x \| A_1^x).$$

By the same argument as [40, Theorem 3] (for completeness, we provide a proof in Appendix B), there exists a predictor $C : \{0,1\}^n \to \{0,1,\perp\}$ that invokes A^x once and satisfies

$$\mathsf{Adv}^{\mathrm{CS}}_{C,f|H} \geq \frac{1}{3} D_{1/2}(A_0^x \| A_1^x) > \frac{\varepsilon}{6}, \tag{26}$$

which contradicts (20). □

Acknowledgements. This work was supported in part by JSPS KAKENHI Grant Numbers 23H00468, 23K17455, and 24H00071.

A Bit-Security Frameworks

As we mentioned in Sect. 1, the bit security for decision game G was first introduced in [32], and an operational framework was later introduced in [39]. For readers' convenience, in this appendix, we review the bit-security frameworks of [32,39] and discuss their equivalence shown in [40]. Since the main focus of this paper is decision games, we only present the formulation of the decision game in the following; see [32,39] for the formulation of the search game.

Let U be the random variable describing the choice of a decision game; for instance, in the indistinguishability game of pseudorandom number generator (PRG) from the true random number generator (TRG), $U = 0$ corresponds to the game played with PRG and $U = 1$ corresponds to the game played with TRG. In the framework of [32], we consider an adversary A that outputs $\perp$ in addition to 0 and 1; the symbol $\perp$ signifies that the adversary has difficulty predicting the value of U and gives up the prediction. For the random variable Y describing the adversary's output, let

$$\alpha_A := \Pr(Y \neq \perp),$$
$$\beta_A := \Pr(Y = U | Y \neq \perp),$$

and define the conditional squared (CS) advantage

$$\mathsf{Adv}^{\mathrm{CS}}_{A,G} := \alpha_A (2\beta_A - 1)^2. \tag{27}$$

Then, the bit security of [32] is defined as[10]

$$\min_A \left\{ \log_2 \left(\frac{s_A}{\mathsf{Adv}^{\mathrm{CS}}_{A,G}} \right) \right\}, \tag{28}$$

[10] In [32], the authors first introduced an advantage using the Shannon entropy and the mutual information; then, in order to justify the definition (28), they discussed that that advantage is approximated by the CS advantage.

where s_A is the cost of the adversary.[11]

In the bit-security framework of [39], in order to define bit-security operationally, we consider an outer adversary B in addition to the inner adversary A that plays the given security game. In the framework, the outer adversary B seeks to increase the success probability of predicting U by invoking the inner adversary A for $N_{A,B}$ times. Then, by integrating the outputs from the invocations of the inner adversary, the outer adversary outputs the predicted value Z. Then, for a prescribed success probability $1-\mu$ (say 0.99), the bit security of decision game G is defined as

$$\mathrm{BS}_G^{\mu} := \min_{A,B} \left\{ \log_2(N_{A,B} \cdot s_A) : \Pr(Z = U) \geq 1 - \mu \right\}. \tag{29}$$

Furthermore, it was shown in [39] that the bit security is characterized by the Rényi advantage up to a constant, i.e.,

$$\mathrm{BS}_G^{\mu} = \min_{A} \left\{ \log_2 s_A + \log_2 \left\lceil \frac{1}{\mathsf{Adv}_{A,G}^{\mathrm{Renyi}}} \right\rceil \right\} + \mathcal{O}(1), \tag{30}$$

where the Rényi advantage is given by the Rényi divergence

$$\mathsf{Adv}_{A,G}^{\mathrm{Renyi}} := D_{1/2}(A_0 \| A_1)$$

for the distributions A_0 and A_1 of adversary for $U = 0$ and $U = 1$, respectively.

At first glance, the bit security defined in (28) and that in (30) (defined via operational formula (29)) are different quantities. However, it was shown in [40] that the two notions of bit security are equivalent in the following sense.

Proposition 3. *For an arbitrary adversary A for a decision game G, it holds that $\mathsf{Adv}_{A,G}^{\mathrm{CS}} \leq 8\mathsf{Adv}_{A,G}^{\mathrm{Renyi}}$. On the other hand, for an arbitrary adversary A satisfying $\mathsf{Adv}_{A,G}^{\mathrm{Renyi}} \leq 1$, there exists an adversary $\tilde{A}$ having the same cost as A such that $\mathsf{Adv}_{A,G}^{\mathrm{Renyi}} \leq 12\mathsf{Adv}_{\tilde{A},G}^{\mathrm{CS}}$.*

By using the conversion of two advantages in Proposition 3, we can argue that the two notions of bit security coincide up to a constant; for more detail, see [40, Section 4]. Since the two notions are equivalent, in the main body of the paper, we focus on the bit security characterized by the Rényi advantage, (30). However, the CS advantage adapted for predictors also plays an important role when we prove the hardcore lemma in Sect. 3. From a technical perspective, it seems that the CS advantage is more suitable for analyzing the performance of algorithms; on the other hand, the Rényi advantage is more convenient for analysis in a certain situation since it satisfies (joint) convexity with respect to the distributions, which is used in the proof of Theorem 1.

Here, we should note that the standard advantage defined by the total variation distance between A_0 and A_1 is unsuitable for evaluating bit security. Notably, as was pointed out in [32] (see also [40, Section 1.3] further discussion), the standard advantage cannot resolve the paradoxical nature of the linear test for the PRG.

[11] In this paper, we focus on the circuit size.

B Proof of (26)

In this appendix, we prove that there exists a predictor $C : \{0,1\}^n \to \{0,1,\perp\}$ that invokes A^x once and satisfies (26). We use the following technical lemma.

Lemma 4. *For given distributions P and Q with $P \ll Q$, we have*

$$D_{1/2}(P\|Q) \le D(P\|Q) \le \sum_{x\in\mathcal{X}^+} \frac{(P(x)-Q(x))^2}{Q(x)}.$$

where $\mathcal{X}^+ = \{x : Q(x) > 0\}$, and $D(P\|Q) = \sum_x P(x)\log(P(x)/Q(x))$ is the KL-divergence.

Proof. The former inequality follows from the fact that the Rényi divergence is monotonically non-decreasing with respect to α and $D(P\|Q) = \lim_{\alpha\to 1} D_\alpha(P\|Q)$. The latter inequality appears in the middle of the proof of [15, Lemma 4.1]. □

Note that, for $y \sim H$, the distribution of the adversary A^x for $u = 0$ instance $(y, f(y))$ and $u = 1$ instance (y, σ) are given by

$$A_0^x(a) = \Pr_{y\sim H}\left(A^x(y, f(y)) = a\right),$$
$$A_1^x(a) = \Pr_{y\sim H}\left(A^x(y, \sigma) = a\right)$$

for $a \in \{0,1,\perp\}$. Note that the support of $(y, f(y))$ is included in the support of (y,σ).[12] Thus, if the adversary A^x outputs a symbol a with positive probability under $u = 0$, then A^x must output a with positive probability under $u = 1$ as well, i.e., $A_0^x \ll A_1^x$.

Let $a^\star \in \{0,1,\perp\}$ be such that $A_1^x(a^\star) > 0$ and

$$\max_{\substack{a\in\{0,1,\perp\}:\\ A_1^x(a)>0}} \frac{(A_0^x(a)-A_1^x(a))^2}{A_1^x(a)} = \frac{(A_0^x(a^\star)-A_1^x(a^\star))^2}{A_1^x(a^\star)}.$$

Then, by Lemma 4, we have

$$D_{1/2}(A_0^x\|A_1^x) \le 3\frac{(A_0^x(a^\star)-A_1^x(a^\star))^2}{A_1^x(a^\star)}. \tag{31}$$

We consider two cases separately.

When $A_0^x(a^\star) \ge A_1^x(a^\star)$ In this case, we consider the following predictor C. First, we sample the uniform random bit σ. Second,

- If $A^x(y,\sigma) = a^\star$, then C outputs σ;
- If $A^x(y,\sigma) \ne a^\star$, then C outputs $\perp$.

[12] Here, the support is the set of realizations that occur with positive probability.

For this predictor, we have

$$\begin{aligned}\Pr_{y\sim H}\left(C(y)\neq\perp\right) &= \Pr_{y\sim H}\left(A^x(y,\sigma)=a^\star\right)\\ &= A_1^x(a^\star)\end{aligned}$$

and

$$\begin{aligned}\Pr_{y\sim H}\left(C(y)=f(y)\right) &= \frac{1}{2}\Pr_{y\sim H}\left(A^x(y,\sigma)=a^\star|\sigma=f(y)\right)\\ &= \frac{1}{2}\Pr_{y\sim H}\left(A^x(y,f(y))=a^\star\right)\\ &= \frac{A_0^x(a^\star)}{2},\end{aligned}$$

which implies

$$\Pr_{y\sim H}\left(C(y)=f(y)\right)-\frac{1}{2}\Pr_{y\sim H}\left(C(y)\neq\perp\right)=\frac{A_0^x(a^\star)-A_1^x(a^\star)}{2}.$$

Thus, the CS advantage of C satisfies (cf. (5))

$$\begin{aligned}\mathsf{Adv}^{\mathrm{CS}}_{C,f|H} &= \frac{(A_0^x(a^\star)-A_1^x(a^\star))^2}{A_1^x(a^\star)}\\ &\geq \frac{1}{3}D_{1/2}(A_0^x\|A_1^x),\end{aligned}$$

where the last inequality follows from (31).

When $A_0^x(a^\star)<A_1^x(a^\star)$ In this case, we consider the following predictor. First, we sample the uniform random bit σ. Second,

- If $A^x(y,\sigma)=a^\star$, then C outputs $\sigma\oplus 1$;
- If $A^x(y,\sigma)\neq a^\star$, then C outputs $\perp$.

For this predictor, we have

$$\begin{aligned}\Pr_{y\sim H}\left(C(y)\neq\perp\right) &= \Pr_{y\sim H}\left(A^x(y,\sigma)=a^\star\right)\\ &= A_1^x(a^\star)\end{aligned}$$

and

$$\begin{aligned}\Pr_{y\sim H}\left(C(y)=f(y)\right) &= \Pr_{y\sim H}\left(\sigma=f(y)\oplus 1, A^x(y,\sigma)=a^\star\right)\\ &= \Pr_{y\sim H}\left(A^x(y,\sigma)=a^\star\right)-\Pr_{y\sim H}\left(\sigma=f(y), A^x(y,\sigma)=a^\star\right)\\ &= A_1^x(a^\star)-\frac{1}{2}\Pr_{y\sim H}\left(A^x(y,\sigma)=a^\star|\sigma=f(y)\right)\\ &= A_1^x(a^\star)-\frac{1}{2}\Pr_{y\sim H}\left(A^x(y,f(y))=a^\star\right)\\ &= A_1^x(a^\star)-\frac{A_0^x(a^\star)}{2},\end{aligned}$$

which implies

$$\Pr_{y\sim H}\big(C(y)=f(y)\big)-\frac{1}{2}\Pr_{y\sim H}\big(C(y)\neq\perp\big)=\frac{A_1^x(a^\star)-A_0^x(a^\star)}{2}.$$

Thus, the CS advantage of C again satisfies

$$\begin{aligned}\mathsf{Adv}^{\mathrm{CS}}_{C,f|H}&=\frac{(A_0^x(a^\star)-A_1^x(a^\star))^2}{A_1^x(a^\star)}\\&\geq\frac{1}{3}D_{1/2}(A_0^x\|A_1^x).\end{aligned}$$

□

C Justification of Update Rule of Algorithm 1

Since we only discuss one iteration of the update, we omit the round number t. For the convenience of notations, in this appendix, we assume that the outputs of function f and circuits are ±1 or 0 with correspondence $0\leftrightarrow+1$, $1\leftrightarrow-1$, and $\perp\leftrightarrow 0$.

Let us start with a review of the update rule of the standard boosting in our terminology (e.g. see [35]). For given distribution P and circuit C_P, the standard boosting algorithm creates the updated distribution as

$$\hat{P}_\gamma(x)=\frac{\exp\big(-\gamma C_P(x)f(x)\big)}{Z(\gamma)}\tag{32}$$

for some $\gamma\in\mathbb{R}$, where

$$Z(\gamma)=\sum_x P(x)\exp\big(-\gamma C_P(x)f(x)\big)$$

is the normalizer. The advantage of C_P for $x\sim P$ can be written as

$$\Pr_{x\sim P}(C_P(x)=f(x))-\Pr_{x\sim P}(C_P(x)=\overline{f(x)}=\sum_x P(x)C_P(x)f(x),$$

where $\overline{f(x)}=-f(x)$. By denoting $q_{+1}=\Pr_{x\sim P}(C_P(x)=f(x))$ and $q_{-1}=\Pr_{x\sim P}(C_P(x)=\overline{f(x)})$, we can rewrite the normalizer $Z(\gamma)$ as

$$Z(\gamma)=q_{+1}e^{-\gamma}+q_{-1}e^{\gamma}.\tag{33}$$

Then, if we set $\gamma^*=\frac{1}{2}\ln\frac{q_{+1}}{q_{-1}}$, then we have

$$\left.\frac{dZ(\gamma)}{d\gamma}\right|_{\gamma=\gamma^*}=\left.\big[-q_{+1}e^{-\gamma}+q_{-1}e^{\gamma}\big]\right|_{\gamma=\gamma^*}=0.$$

By noting these facts, we can compute the advantage of C_P for $x \sim \hat{P}_\gamma$ as

$$\sum_x \hat{P}_\gamma(x) C_P(x) f(x) = \frac{1}{Z(\gamma)} \sum_x P(x) \exp\big(-\gamma C_P(x) f(x)\big) C_P(x) f(x)$$
$$= -\frac{1}{Z(\gamma)} \frac{dZ(\gamma)}{d\gamma}.$$

Thus, if we set $\gamma = \gamma^*$ in the update rule, the advantage of C_P for $x \sim \hat{P}_{\gamma^*}$ is 0. Let us denote $\Delta = q_{+1} - q_{-1}$. Then, we have

$$\gamma^* = \frac{1}{2} \ln(1 + \Delta) - \frac{1}{2} \ln(1 - \Delta) \simeq \Delta$$

for small Δ. In fact, the approximate value Δ has been used in the update of the standard boosting algorithm in the literature (e.g. [3,24]).

Next, let us consider the boosting with $\perp$. We use the same notations as above, and the update rule itself is also given by (32). However, note that C_P may also output 0 (corresponding to $\perp$). Let $q_0 = \Pr_{x \sim P}(C_P(x) = 0)$. Note that the CS advantage of C_P for $x \sim P$ can be written as

$$\mathsf{Adv}^{\mathrm{CS}}_{C_P, f|P} = \frac{(q_{+1} - q_{-1})^2}{1 - q_0}.$$

The normalizer can be written as

$$Z(\gamma) = \sum_x P(x) \exp\big(-\gamma C_P(x) f(x)\big)$$
$$= q_{+1} e^{-\gamma} + q_{-1} e^{\gamma} + q_0.$$

Thus, if we set $\gamma^* = \frac{1}{2} \ln \frac{q_{+1}}{q_{-1}}$, then we again have

$$\left.\frac{dZ(\gamma)}{d\gamma}\right|_{\gamma=\gamma^*} = \left[-q_{+1} e^{-\gamma} + q_{-1} e^{\gamma}\right]\Big|_{\gamma=\gamma^*} = 0.$$

By denoting

$$\hat{\alpha}_\gamma = \Pr_{x \sim \hat{P}_\gamma}(C_P(x) \neq 0),$$
$$\hat{\Delta}_\gamma = \Pr_{x \sim \hat{P}_\gamma}(C_P(x) = f(x)) - \Pr_{x \sim \hat{P}_\gamma}(C_P(x) = \overline{f(x)}),$$

the CS advantage of C_P for $x \sim \hat{P}_\gamma$ can be written as $\mathsf{Adv}^{\mathrm{CS}}_{C_P, f|\hat{P}_\gamma} = \frac{\hat{\Delta}_\gamma^2}{\hat{\alpha}_\gamma}$. Note that

$$\hat{\Delta}_\gamma = \sum_x \hat{P}_\gamma(x) C_P(x) f(x)$$
$$= \frac{1}{Z(\gamma)} \sum_x P(x) \exp\big(-\gamma C_P(x) f(x)\big) C_P(x) f(x)$$
$$= -\frac{1}{Z(\gamma)} \frac{dZ(\gamma)}{d\gamma}.$$

Thus, if we set $\gamma = \gamma^*$ in the update rule, the CS advantage of C_P for $x \sim \hat{P}_{\gamma^*}$ satisfies $\mathsf{Adv}^{\mathrm{CS}}_{C_P, f|\hat{P}_{\gamma^*}} = 0$. By denoting $\alpha = 1 - q_0$ and $\Delta = q_{+1} - q_{-1}$, we have

$$\begin{aligned}\gamma^* &= \frac{1}{2}\ln\left(\frac{1}{2}(1-q_0) + \frac{\Delta}{2}\right) - \frac{1}{2}\ln\left(\frac{1}{2}(1-q_0) - \frac{\Delta}{2}\right) \\ &= \frac{1}{2}\ln\left(1 + \frac{\Delta}{\alpha}\right) - \frac{1}{2}\ln\left(1 - \frac{\Delta}{\alpha}\right) \\ &\simeq \frac{\Delta}{\alpha}.\end{aligned}$$

for small $\frac{\Delta}{\alpha}$. Note that $\frac{\Delta}{4\alpha}$ is the multiplicative weight used in Algorithm 1. Thus, the update rule in Algorithm 1 can be regarded as an approximate version of the rule such that the CS advantage of the current weak learner for the updated distribution is zero.

References

1. Aslam, J.A.: Improving algorithms for boosting. In: Proceedings of the 13th Annual Conference on Computational Learning Theory, pp. 200–207 (2000)
2. Bader, C., Jager, T., Li, Y., Schäge, S.: On the impossibility of tight cryptographic reductions. In: Fischlin, M., Coron, J. (eds.) Advances in Cryptology - EUROCRYPT 2016. Lecture Notes in Computer Science, vol. 9666, pp. 273–304. Springer (2016). https://doi.org/10.1007/978-3-662-49896-5_10
3. Barak, B., Hardt, M., Kale, S.: The uniform hardcore lemma via approximate Bregman projections. In: Proceedings of the 2009 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 1193–1200 (2009)
4. Bellare, M., Ristenpart, T.: Simulation without the artificial abort: Simplified proof and improved concrete security for Water's IBE scheme. In: Advances in Cryptology - EUROCRYPT 2009, 28th Annual International Conference on the Theory and Applications of Cryptographic Techniques. Lecture Notes in Computer Science, vol. 5479, pp. 407–424. Springer (2009). https://doi.org/10.1007/978-3-642-01001-9_24
5. Csiszár, I., Shields, P.C.: Information theory and statistics: a tutorial. Found. Trends Commun. Inf. Theory **1**(4) (2004). https://doi.org/10.1561/0100000004
6. Diemert, D., Jager, T.: On the tight security of TLS 1.3: Theoretically sound cryptographic parameters for real-world deployments. J. Cryptol. **34**(3), 30 (2021). https://doi.org/10.1007/s00145-021-09388-x
7. Dodis, Y., Impagliazzo, R., Jaiswal, R., Kabanets, V.: Security amplification for interactive cryptographic primitives. In: Reingold, O. (ed.) Theory of Cryptography, 6th Theory of Cryptography Conference, TCC 2009, San Francisco, CA, USA, March 15-17, 2009. Proceedings. Lecture Notes in Computer Science, vol. 5444, pp. 128–145. Springer (2009). https://doi.org/10.1007/978-3-642-00457-5_9
8. van Erven, T., Harremoës, P.: Rényi divergence and Kullback-Leibler divergence. IEEE Trans. Inform. Theory **60**(7), 3797–3820 (2014)
9. Goldmann, M., Håstad, J., Razborov, A.: Majority gates vs. general weighted threshold gates. Comput. Complex. **2**, 277–300 (1992)
10. Goldreich, O., Nisan, N., Wigderson, A.: On Yao's XOR lemma. Electronic Colloquium on Computational Complexity (March 1995)

11. Goldreich, O.: The Foundations of Cryptography - Volume 1: Basic Techniques. Cambridge University Press (2001). https://doi.org/10.1017/CBO9780511546891, http://www.wisdom.weizmann.ac.il/%7Eoded/foc-vol1.html
12. Goldreich, O.: On security preserving reductions - revised terminology. In: Goldreich, O. (ed.) Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation, Lecture Notes in Computer Science, vol. 6650, pp. 540–546. Springer (2011)
13. Goldreich, O., Impagliazzo, R., Levin, L.A., Venkatesan, R., Zuckerman, D.: Security preserving amplification of hardness. In: 31st Annual Symposium on Foundations of Computer Science, St. Louis, Missouri, USA, October 22-24, 1990, Volume I, pp. 318–326. IEEE Computer Society (1990). https://doi.org/10.1109/FSCS.1990.89550
14. Goldreich, O., Nisan, N., Wigderson, A.: On Yao's XOR-lemma. In: Goldreich, O. (ed.) Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation, Lecture Notes in Computer Science, vol. 6650, pp. 273–301. Springer (2011). https://doi.org/10.1007/978-3-642-22670-0_23
15. Götze, F., Sambale, H., Sinulis, A.: Higher order concentration for functions of weakly dependent random variables. Electron. J. Probab. **24**(85), 1–19 (2019)
16. Haitner, I., Reingold, O., Vadhan, S.P.: Efficiency improvements in constructing pseudorandom generators from one-way functions. SIAM J. Comput. **42**(3), 1405–1430 (2013). https://doi.org/10.1137/100814421
17. Hast, G.: Nearly one-sided tests and the Goldreich-Levin predicate. J. Cryptol. **17**, 209–229 (2004)
18. Håstad, J., Impagliazzo, R., Levin, L.A., Luby, M.: A pseudorandom generator from any one-way function. SIAM J. Comput. **28**(4), 1364–1396 (1999). https://doi.org/10.1137/S0097539793244708
19. Hatano, K., Warmuth, M.K.: Boosting versus covering. In: Advances in Neural Information Processing Systems. vol. 16 (2003)
20. Hatano, K., Watanabe, O.: Learning r-of-k functions by boosting. In: Proceedings of the 15th International Conference on Algorithmic Learning Theory, pp. 114–126 (2004)
21. Herzberg, A., Luby, M.: Pubic randomness in cryptography. In: Brickell, E.F. (ed.) Advances in Cryptology - CRYPTO '92, 12th Annual International Cryptology Conference, Santa Barbara, California, USA, August 16-20, 1992, Proceedings. Lecture Notes in Computer Science, vol. 740, pp. 421–432. Springer (1992). https://doi.org/10.1007/3-540-48071-4_29
22. Holenstein, T.: Key agreement from weak bit agreement. In: Proceedings of the 37th Annual ACM Symposium on Theory of Computing (STOC '05), pp. 664–673. ACM Press (2005)
23. Impagliazzo, R.: Hard-core distribution for somewhat hard problems. In: Proceedings of the 36th Annual IEEE Symposium on Foundations of Computer Science (FOCS '95), pp. 538–545 (1995)
24. Kale, S.: Boosting and hard-core set constructions: a simplified approach (2007), electronic Colloquium on Computational Complexity (ECCC), Report No. 131
25. Klivans, A.R., Servedio, R.A.: Boosting and hard-core set construction. Mach. Learn. **51**, 217–238 (2003)
26. Lanzenberger, D., Maurer, U.: Direct product hardness amplification. In: Nissim, K., Waters, B. (eds.) Theory of Cryptography - 19th International Conference, TCC 2021, Raleigh, NC, USA, November 8-11, 2021, Proceedings, Part II. Lecture Notes in Computer Science, vol. 13043, pp. 605–625. Springer (2021)

27. Lee, K.: Bit security as cost to demonstrate advantage. IACR Commun. Cryptol. **1**(1) (2024). https://doi.org/10.62056/an5txol7
28. Li, B., Micciancio, D., Schultz, M., Sorrell, J.: Securing approximate homomorphic encryption using differential privacy. In: Dodis, Y., Shrimpton, T. (eds.) Advances in Cryptology - CRYPTO 2022. Lecture Notes in Computer Science, vol. 13507, pp. 560–589. Springer (2022). https://doi.org/10.1007/978-3-031-15802-5_20
29. Luby, M.: Pseudorandomness and cryptographic applications. Princeton University Press, Princeton computer science notes (1996)
30. Maurer, U., Tessaro, S.: A hardcore lemma for computational indistinguishability: Security amplification for arbitrary weak PRGs with optimal stretch. In: TCC 2010. Lecture Notes in Computer Science, vol. 5078, pp. 237–254. Springer (2010)
31. Maurer, U.M., Tessaro, S.: Computational indistinguishability amplification: Tight product theorems for system composition. In: Halevi, S. (ed.) Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings. Lecture Notes in Computer Science, vol. 5677, pp. 355–373. Springer (2009). https://doi.org/10.1007/978-3-642-03356-8_21
32. Micciancio, D., Walter, M.: On the bit security of cryptographic primitives. In: Nielsen, J.B., Rijmen, V. (eds.) Advances in Cryptology - EUROCRYPT 2018. Lecture Notes in Computer Science, vol. 10820, pp. 3–28. Springer (2018). https://doi.org/10.1007/978-3-319-78381-9_1
33. Morgan, A., Pass, R.: On the security loss of unique signatures. In: Theory of Cryptography. Theory of Cryptography, vol. 11239, pp. 507–536. Springer (2018). https://doi.org/10.1007/978-3-030-03807-6_19
34. Myers, S.A.: Efficient amplification of the security of weak pseudo-random function generators. J. Cryptol. **16**(1), 1–24 (2003). https://doi.org/10.1007/s00145-002-0007-1
35. Schapire, R.E., Freund, Y.: Boosting: Foundations and Algorithms. MIT Press (2012)
36. Schapire, R.E., Singer, Y.: Improved boosting algorithm using confidence-rated predictions. Mach. Learn. **37**(3), 297–336 (1999)
37. Vadhan, S., Zheng, C.J.: A uniform min-max theorem with applications in cryptography. In: Advances in Cryptography–CRYPTO '13. Lecture Notes in Computer Science, vol. 8042, pp. 93–110. Springer (2013)
38. Vadhan, S., Zheng, C.J.: Characterizing pseudoentropy and simplifying pseudorandom generator constructions. STOC '12, Association for Computing Machinery, New York, NY, USA (2012). https://doi.org/10.1145/2213977.2214051
39. Watanabe, S., Yasunaga, K.: Bit security as computational cost for winning games with high probability. In: Tibouchi, M., Wang, H. (eds.) Advances in Cryptology - ASIACRYPT 2021. Lecture Notes in Computer Science, vol. 13092, pp. 161–188. Springer (2021)
40. Watanabe, S., Yasunaga, K.: Unified view for notions of bit security. In: Guo, J., Steinfeld, R. (eds.) Advances in Cryptology - ASIACRYPT 2023. Lecture Notes in Computer Science, vol. 14443, pp. 361–389. Springer (2023)
41. Wegener, I.: The Complexity of Boolean Functions. Wiley (1991)

Bit Security: Optimal Adversaries, Equivalence Results, and a Toolbox for Computational-Statistical Security Analysis

Daniele Micciancio(✉) and Mark Schultz-Wu

University of California, San Diego, USA
daniele@cs.ucsd.edu, mdschult@ucsd.edu

Abstract. We investigate the notion of bit-security for decisional cryptographic properties, as originally proposed in (Micciancio & Walter, Eurocrypt 2018), and its main variants and extensions, with the goal clarifying the relation between different definitions, and facilitating their use. Specific contributions of this paper include: (1) identifying the optimal adversaries achieving the highest possible MW advantage, showing that they are deterministic and have a very simple threshold structure; (2) giving a simple proof that a competing definition proposed by (Watanabe & Yasunaga, Asiacrypt 2021) is actually equivalent to the original MW definition; and (3) developing tools for the use of the extended notion of computational-statistical bit-security introduced in (Li, Micciancio, Schultz & Sorrell, Crypto 2022), showing that it fully supports common cryptographic proof techniques like hybrid arguments and probability replacement theorems. On the technical side, our results are obtained by introducing a new notion of "fuzzy" distinguisher (which we prove equivalent to the "aborting" distinguishers of Micciancio and Walter), and a tight connection between the MW advantage and the *Le Cam* metric, a standard quantity used in statistics.

Keywords: Bit security · Computational security · Statistical security · Le Cam distance

1 Introduction

The level of security provided by a cryptographic construction is customarily measured in "bits". The intuition is that breaking an application offering "n bits of security" should have a cost[1] comparable to mounting a key recovery attack on an ideal cryptographic function with a key space of size 2^n. Formalizing this intuition is not entirely trivial, because cryptographic attacks often exhibit a

[1] Various measures of cost have been considered, and the reader is referred to [2, Appendix B] for a discussion. For simplicity, in this paper we identify the cost of an attack with its running time.

E. Boyle and M. Mahmoody (Eds.): TCC 2024, LNCS 15365, pp. 224–254, 2025.
https://doi.org/10.1007/978-3-031-78017-2_8

trade-off between the cost (e.g., the running time T_A) of the attack, and its success probability ϵ_A. For (verifiable) search problems, like forging digital signatures, it is well established[2] that bit security can be defined as the quantity $\log_2(T_A/\epsilon_A)$, minimized over all possible adversaries A. However, the situation for decision problems (like indistinguishability of ciphertexts, zero knowledge, pseudorandomness, etc.) is far less clear. We recall that in a decision game the goal of the adversary is to distinguish between two distributions X_b for $b \in \{0,1\}$. So, a naive approach to measure security could be to mimic the definition for search problems, and replace the quantity $\log_2(T_A/\epsilon_A)$ with $\log_2(T_A/\delta_A)$, where $\delta_A = 2\epsilon_A - 1$ is the advantage (over a random choice) of guessing the bit b. But it is well known that this naive definition leads to paradoxical situations, where for example [3] an algorithm G is deemed more secure (i.e., it is attributed a higher level of bit security) as a pseudorandom generator than as a one-way function. This is at odds with cryptographic intuition because pseudorandomness is a stronger security requirement than one-wayness. (See [10] and references therein for a detailed discussion of this and other problematic examples.)

During the last few years, several papers have investigated the problem of giving meaningful definitions of bit security [6,8,10,14,15], or using them to give a tight security analysis of cryptographic primitives (e.g., see [1,8]). A satisfactory definition of bit security for decision games was first proposed by Micciancio and Walter in [10]. A key element of their definition is to consider attackers that may output either a bit $b \in \{0,1\}$ (indicating a decision between X_0 and X_1) or a special "don't know" symbol $\perp$. Interestingly, [10] shows that this simple extension of traditional adversaries, together with an appropriate definition of advantage (already used by [7] in a different context,) allows to resolve all the previously mentioned paradoxes, and argues (by means of examples) that this is the right definition of bit security.[3] Since then, a number of alternative definitions have appeared [6,8,14,15], with various motivations. Watanabe and Yasunaga [14] proposed a competing framework to define bit security that directly admits what they call an "operational interpretation", and later argued [15] that it is actually equivalent to the original MW definition [10]. A seemingly attractive feature of their definition is that it only requires standard (non-aborting) adversaries with output in $\{0,1\}$. A variant of their definition that (similarly to [10]) interpolates between search and decision problems is given in [6]. In a different and orthogonal direction, Li, Micciancio, Schultz and Sorrell [8] extend the MW definition to encompass both computational and statistical security. Informally, statisti-

[2] This is justified by the fact that one can repeat the attack $O(1/\epsilon)$ times to make the success probability arbitrarily close to 1.

[3] This is at least for search (key recovery) and decision problems. The work [10] also proposes a more general definition based on information theory that interpolates between search and decision problems (e.g., encompassing password recovery problems with a polynomially large set of secrets,) but the corresponding notion of bit security for intermediate cases is largely unexplored. In this paper, we focus on the special case of decision problems which is the most relevant to cryptography. For search problems, the general bit-security definition of [10] reduces to $\log_2 T_A/\epsilon_A$, which is standard, and is adopted in this paper too.

cal security provides a strong measure of security even against computationally unbounded adversaries. When achievable, statistical security has the advantage of being easier to anaylze, and not requiring any computational assumptions. In practice, when setting parameters and optimizing efficiency, it is common to require lower levels of statistical bit security s, than computational bit security c. For example, $s = 80$ is usually considered more than acceptable, while computational security typically requires $c \geq 128$ or even higher values to anticipate possible improvements in the computational complexity of attacks. Li at al. [8] define (c, s)-security as satisfied by a protocol that provides *either* c bits of computational security, *or* s bits of statistical security against any possible attack. We remark that a protocol can admit both attacks with running time much less than 2^c (as long as their advantage is less than 2^{-s}) and (different) attacks achieving advantage very close to 1 (as long as their running time is higher than 2^c). In other words, a (c, s)-secure protocol can achieve neither c-bits of computational security nor s-bits of statistical security. Still, morally, it provides an acceptable level of security wherever s-bit statistical security and c-bit computational security are considered individually adequate. The advantage of (c, s)-security is that it allows to seamlessly combine statistical and computational cryptographic primitives (something very common in practice) and still be able to formally quantify the security level of an application. However, the notion of (c, s)-security has not been further explored, and, despite its potential usefulness, it has seen little adoption due to the lack of tools to simplify its usage.

Our Contributions and Techniques. In this work, we examine the bit security definitions of [8,10,14], proving structural results about optimal (statistical) adversaries, clarifying the relation between the MW and WY bit security definitions, and then applying these results to the recent notion of (c, s)-bit security. Our main contributions, described in more details in the next subsections, can be summarized as follows:

- We characterize the MW adversaries achieving the optimal (statistical) bit-security advantage. Specifically, we show that these adversaries may be assumed to be deterministic (Corollary 1) and have a simple "threshold" structure (Theorems 3).
- We show (Theorem 4) that the WY notion of bit security is equivalent to the original MW bit security definition. In other words, the definition put forward in [14] is not a new security notion, but a different formulation of MW bit-security which, potentially, may be more convenient in some settings. We remark that a proof of this equivalence was already given in [15], but, as we are going to describe, that proof contained a gap. We clarify the relation between the two definitions by filling the gap and also giving a simpler proof of the equivalence.
- Despite the fact that the WY definition only uses traditional (non-aborting) adversaries, we show (Theorem 5) that the natural "maximum likelihood" distinguisher can offshoot the correct bit security level by a large margin. So, the advantages of using standard (non-aborting) adversaries in the characterization of bit security put forward in [14] are unclear.

– We show that common proof techniques widely used in the analysis of cryptographic protocols can be extended to work with the more general notion of computational-statistical security from [8]. Specifically, we show that (c, s)-security fully supports the use of hybrid arguments (Theorem 6) and probability substitution (Theorem 7).

On the technical side, many of our results rely on a new class of adversaries that further extends the MW (aborting) adversaries, and that may be of independent interest. Specifically, we make use of adversaries (for decision games) that may output not just $0, 1$ (representing a high confidence decision) or $\perp$ (representing no confidence), but an arbitrary value $\sigma \in [-1, 1]$, with the sign $\sigma/|\sigma| \in \{-1, 1\}$ representing the decision, and the magnitude $|\sigma| \in [0, 1]$ the confidence level that can vary continuously from 0 (no confidence) to 1 (perfect confidence). Interestingly, we show (Theorem 1) that these "fuzzy" adversaries still define precisely the same notion of bit security as the original MW "aborting" adversaries. The robustness of the notion of bit security with respect to such extensions further supports the use of [10] as the standard notion of bit security. Still, our equivalent definition using fuzzy adversaries with output in the continuous interval $[-1, 1]$ supports the use of analytical techniques, and it is useful to prove some of the results in this paper. We believe that the characterization of bit security in terms of these more general fuzzy adversaries may find other applications, and is of independent interest.

Related Work. As mentioned, our work directly builds on the bit security frameworks of [10,14], so is directly related to these works. Our work is also tangentially related to the bit security framework of [6], though this work mostly focuses on generalizing (a variant of) the framework of [14] to non-decision games, whereas we focus on decision games. Our work on the optimal adversary for the MW advantage is additionally related to the notion of (binary) hypothesis testing with an aborting option, see for example [4], though the measure optimized in that work does not appear to be related to the MW advantage. The similarity between our work and binary hypothesis testing with a rejection option is perhaps more obvious from [5, Section 4], where (in a slightly different setting) optimality of threshold distinguishers was also highlighted. Still in relation to hypothesis testing, we discuss the implications of [11] to the WY formulation of bit security.

Paper Organization. The rest of this paper is organized as follows. In the rest of this section we give a more detailed, still informal, description of our results and techniques. Section 2 defines the notation and preliminary results used in this paper. In Sect. 3 we formally define fuzzy adversaries, establish their equivalence (in both the computational and statistical setting) with the aborting adversaries of [10], and then use them to investigate the structure of the (statistical) MW adversaries achieving the optimal advantage. In Sect. 4, we explore the WY bit security definition and its equivalence with the MW bit security. Finally, in Sect. 5, we build our toolbox for the use of (c, s)-security in the analysis of

cryptographic protocols, establishing the validity of hybrid arguments and probability replacement theorems. Section 6 concludes with some final remarks and open problems.

1.1 The Micciancio-Walter Advantage

Consider the problem of distinguishing between two distributions $\mathcal{X} = (X_0, X_1)$ over a set Ω. (Everything applies more generally to the case of more complex decision games where an adversary interacts with one of two oracles.) Micciancio and Walter (following [7]) define the advantage of an "aborting" adversary $A : \Omega \to \{0, 1, \perp\}$ as

$$\mathsf{adv}^{\mathsf{MW}}_{\mathcal{X}}(A) = \frac{(\beta_A - \bar{\beta}_A)^2}{\beta_A + \bar{\beta}_A}, \tag{1}$$

where $\beta_A = \Pr[A(x_b) = b]$ and $\bar{\beta}_A = \Pr[A(x_b) = 1 - b]$ are the probability that A outputs the correct or incorrect bit, respectively, when $b \in \{0, 1\}$ is chosen at random and $x_b \leftarrow X_b$. For traditional (non-aborting) adversaries with output in $\{0, 1\}$, we have $\beta + \bar{\beta} = 1$, and it is well known (and quite intuitive) that the best advantage is achieved by an adversary $A(x)$ that on input a sample $x \in \Omega$, outputs the bit $b \in \{0, 1\}$ for which $\Pr[X_b = x]$ is highest. Moreover, the resulting optimal advantage equals precisely the square $\Delta_{\mathsf{SD}}(X_0, X_1)^2$ of the statistical distance between the two distributions. This allows to easily compute the bit security of $\mathcal{X}$ whenever the probability distributions are efficiently computable. This is a common scenario in cryptography, where, for example X_0 may be an ideal probability distribution used in the theoretical analysis of a cryptographic scheme (e.g., a discrete gaussian distribution in lattice-based cryptography) and X_1 is an approximate (more efficiently samplable) version of X_0 used when implementing the algorithm in practice (e.g. using floating point numbers). In fact, this was precisely the motivation in [9,10].

However, once the adversary is allowed to output $\perp$, it is no longer clear how to determine an optimal adversarial strategy, even when the probability distributions X_0, X_1 are efficiently computable. For example, while intuitively it is clear that the adversary $A(x)$ should output $\perp$ (and express low confidence) when the probabilities $p_0 = \Pr\{X_0 = x\}$ and $p_1 = \Pr\{X_1 = x\}$ are very close to each other, it is unclear how close is "very close" or even how to measure closeness, e.g., by $|p_0 - p_1|$, p_0/p_1, or some other function (of x) that depends on the global properties of X_0 and X_1. One of our main goals is to characterize the optimal aborting adversarial strategies, both to improve our understanding of the MW bit security definition, and offer a simple tool for the computation of the bit-security distance between specific distributions that may occur in practice.

To this end, we first show that one can equivalently phrase the study of aborting adversaries in terms of the class of *fuzzy adversaries* $\mathcal{A}_{\approx} := \{\tilde{A} \mid \tilde{A} : \Omega \to [-1, 1]\}$. These adversaries' output $y = A(x)$ represents not only a guess of which distribution they are given (via $y/|y| \in \{\pm 1\}$), but also a *confidence level* $|y| \in [0, 1]$. One then measures the advantage of fuzzy adversaries with a "continuous" analogue of (refeq:aborting-advantage) (Definition 6), which we write as

$\mathsf{adv}_{\mathcal{X}}^{\mathsf{MW},\approx}(\tilde{A})$. We prove equivalence (Theorem 10) by giving efficient, advantage-preserving transformations between the two classes of adversaries. This shows that, when maximized over the set of all possible adversaries, $\mathsf{adv}^{\mathsf{MW}}(A)$ and $\mathsf{adv}^{\mathsf{MW},\approx}(A)$ are equivalent. Moreover, since the transformations between fuzzy and aborting adversaries used in our proofs also preserve the adversary's running time, they also establish the equivalence between the corresponding notions of *computational* (and, looking forward, *computational-statistical*) bit security.

We then prove a number of useful properties for aborting and fuzzy adversaries. For example, we show that the MW advantage is a convex function of randomized aborting adversary. As a simple corollary, we derive that the optimal advantage is always achieved by a deterministic aborting adversary (Corollary 1), obtained by fixing the randomness of the probabilistic adversary. This fact, while intuitively obvious[4] and often considered a *folk theorem*, is not generally true, and we give an explicit counterexample demonstrating how it can fail. (See Lemma 12.)

Next we dig deeper into the structure of the optimal fuzzy adversary when probabilities are efficiently computable (or adversaries are computationally unbounded.) We already established that optimal fuzzy adversaries may be assumed to always declare (for any given input) either *full* confidence or *no* confidence at all in their decision. Here we characterize when optimal fuzzy adversaries have full confidence, i.e., output ± 1 instead of 0. Specifically, we show that the optimal adversary must have confidence 0 precisely when the quantity $|\log \Pr\{X_0 = x\} / \Pr\{X_1 = x\}|$ is below a certain threshold $\tau \in [0, \log 3]$, which is a simple function of the adversary's conditional success probability (Theorem 3).

1.2 Watanabe-Yasunaga Bit Security

We next investigate the optimal adversary for Watanabe-Yasunaga Bit Security. On the technical side, our work here is less novel, as information theorists had alrady identified [11] a natural choice of adversaries that fit our purposes. Before discussing the precise results, we briefly provide some background. Watanabe-Yasunaga Bit security (as originally defined in [14]) is specified in terms of an "inner" adversary A that on input a sample $x \leftarrow X_b$, outputs either 0 or 1. This adversary is run n times $y_1 = A(x_1), \ldots, y_n = A(x_n)$ on independent samples $x_i \leftarrow X_b$ all chosen from the same unknown distribution. The number of samples n is chosen large enough so that the value of the bit b can be determined with very high probability $\mu \approx 1$ (say, $\mu \geq 0.99$) based on the output values $y_1, \ldots, y_n$. So, the total running time is given by $n \cdot T_A$, and [14] defines the bit security to be $\log_2(n \cdot T_A)$, minimized over all inner adversaries A and number of repetitions n such that $\mu \geq 0.99$. They also show that this quantity can be equivalently estimated as $\log(T_A / \mathcal{R}_{1/2}(A_0, A_1))$ where $\mathcal{R}_{1/2}$ is the Renyi divergence of order $1/2$, and $A_b = A(X_b) \in \{0, 1\}$ is the Bernoulli random variable defined by the output of the adversary on input a sample from X_b.

[4] We believe that this is the case because we tend to give convexity for granted.

At this point, it is natural to ask:

- What is the relation between the MW and WY bit security?
- What is the optimal adversary $A(x) \in \{0, 1\}$ for the WY definition?

Notice that since the WY adversaries always output either 0 or 1, they are potentially easier to use, as the attacker does not have to choose whether or not to abort.

Regarding the first question, [14] proves only the inequality[5] $MW \leq WY$, showing that WY is a more conservative notion of bit security, and leaving a more precise comparison as an open problem. In a follow-up paper [15] the same authors claimed the equivalence between MW and WY (up to an additive constant), but with a catch. Technically, they prove the equivalence between MW and WY bit security for the same class of *aborting* adversaries (with output in $\{0, 1, \perp\}$) introduced in [10]. Then, they claim equivalence with the original WY definition by informally stating that the definition in [14] does not explicitly depend on the size of the co-domain[6] of the adversary A. However, the justification is incorrect because the Renyi divergence $\mathcal{R}_{1/2}(A(X_0), A(X_1))$ implicitly depends on the size of the co-domain of A. Despite this gap in the proof, we show that the main claim of [15] (about the equivalence of MW and WY bit security) is correct, and in the process we give a simpler and tighter proof of this fact. (Theorem 4.)

So, at this point, the WY notion of bit security can be considered an alternative characterization of the MW bit security, rather than a new definition, and the question is whether this alternative characterization can help in evaluating the bit security of decision problems. One seemingly attractive feature of the WY is that it uses standard adversaries $A(x)$ which always output $0, 1$. This is because for these adversaries there is a particularly natural attack, that on input a sample x outputs the bit b for which the probability $\Pr[X_b = x]$ is highest. However, this does not seem to help in evaluating the bit security using the WY characterization: we show (Theorem 5) that there exist distinguishing games where this natural adversarial strategy yields bit security estimates that are far from optimal.[7]

1.3 Computational/Statistical Bit Security

Finally, we investigate the definition of (c, s)-bit security proposed in [8], to extend MW bit security to encompass both computational and statistical security. Recall that the MW (computational) bit security of a problem is the largest c such that $T(A)/\mathsf{adv}^{\mathsf{MW}}_{\mathcal{X}}(A) \geq 2^c$ for all adversaries A. Similarly, statistical security can be defined as the largest s such that $1/\mathsf{adv}^{\mathsf{MW}}_{\mathcal{X}}(A) \geq 2^s$ for all adversaries

[5] Here and elsewhere we use MW and WY as a shorthand for the number of bits of security as computed according to the respective definitions.

[6] By *co-domain* we mean the set of possible outputs of the adversary, i.e., $|\{0, 1\}| = 2$ for traditional distinguishers and $|\{0, 1, \perp\}|$ for aborting adversaries.

[7] Specifically, the estimates are twice as high as the optimal, correct value. Recall that bit security (roughly) measures the *exponent* of the running time of the adversary. So, a multiplicative factor in bit security estimation is quite large.

A, where this time the running time of A is ignored. Li et al. [8] define a protocol to be (c, s)-secure if for any adversary A

$$\text{either} \quad \frac{T(A)}{\mathsf{adv}^{\mathsf{MW}}_{\mathcal{X}}(A)} \geq 2^c \qquad \text{or} \quad \frac{1}{\mathsf{adv}^{\mathsf{MW}}_{\mathcal{X}}(A)} \geq 2^s.$$

As explained in the introduction, a protocol satisfying this definition seems to provide an adequate level of security whenever computational security and statistical security are considered individually acceptable. Here we point out that a protocol can offer neither c bits of computational security nor s bits of statistical security, and still be (c, s)-secure. Consider for example a protocol such that there exist a very efficient adversary A_c with running time $T(A_c) = 1$ that achieves MW advantage 2^{-s}, and some other adversary A_s with very large running time $T(A_s) \geq 2^c$ that achieves MW advantage ≈ 1. Then, the protocol is neither computationally nor statistically secure because A_c breaks computational security (for $s < c$), and A_s breaks statistical security. So, (c, s)-security is strictly weaker than both c-bits computational security, and s-bits of statistical security. In fact, one should expect this to be the case in any application that makes use of both computational and statistical security primitives, as an adversary can choose to attack the application by trying to break either one or the other type of primitives.

While (c, s)-bit security was introduced in [8] (and successfully used to analyze a practical protocol), this was done via direct manipulation of the definition. In this paper we establish a tight connection between the MW advantage $\mathsf{adv}^{\mathsf{MW}}_{\mathcal{X}}(A)$ and a standard distance measure used in statistics: the (squared) *Le Cam distance* $\Delta^2_{\mathsf{LC}}(A(X_0), A(X_1))$ between the adversary's output probability distributions. Then, we use this connection to prove several useful properties of the (c, s)-bit security which support two of the most common proof techniques in cryptography:

- The "hybrid argument" (see Theorem 6 for the formal statement): consider a sequence of distributions $X_0, \ldots, X_k$. If the game defined by any pair of neighboring distributions (X_{i-1}, X_i) is (c, s)-secure, then the game defined by the extremal distributions (X_0, X_k) is also (c', s')-secure, for $c' \approx c - \log k$ and $s' \approx s - \log k$.
- The "distribution replacement" theorem (see Theorem 7 for the formal statement): Consider a decision game (X_0^Y, X_1^Y) parameterized by a distribution Y. If distinguishing between (Y, Y') is (c, s)-secure, and (X_0^Y, X_1^Y) is (c, s)-secure, then $(X_0^{Y'}, X_1^{Y'})$ is also (c', s')-secure, for $c' \approx c$ and $s' \approx s$.

Our results improve or extend previous work. For example, [10] had already proved a hybrid theorem for computational bit security, and hybrid theorems for statistical bit security are essentially a form of (pythagorean) triangle inequality for the associated distance functions between distributions. The novelty here is to establish the validity of hybrid arguments in the more general computational-statistical setting, where each pair of neighboring distributions (X_{i-1}, X_i) may be neither computationally nor statistically indistinguishable.

Distribution replacement theorems for bit security were previously proved in [10,16], but only for the setting where (X_0^Y, X_1^Y) are computationally close and (Y, Y') are *statistically* close (either in the max-log or Hellinger distance.) Our theorem allows both (X_0^Y, X_1^Y) and (Y, Y') to be close in the much weaker sense of computational-statistical bit security.

Both types of techniques are cornerstones for the modular analysis of complex cryptographic protocols that combine several cryptographic primitives. Our results support the uniform use of computational-statistical bit-security to analyze both the final application and its building blocks, including neighboring hybrids (X_{i-1}, X_i) and probability replacements (Y, Y'). Moreover, they support the seamless combination of computational and statistical security primitives, while at the same time offering tight security estimates, which, before our work, could only be done either informally or using ad-hoc methods. The connection with the Le Cam metric, which underlies our proofs, is also of independent interest, and may find other applications.

2 Preliminaries

We will make use of the following variant of the Cauchy-Schwarz inequality.

Lemma 1 (Bergström's Inequality). *For any real numbers $a_1, \ldots, a_n$, and positive reals $b_1, \ldots, b_n$, we have that*

$$\frac{(\sum_{i\in[n]} a_i)^2}{\sum_{i\in[n]} b_i} \leq \sum_{i\in[n]} \frac{a_i^2}{b_i}.$$

Proof. Rearrange the Cauchy-Schwarz inequality to $\frac{\langle c,d\rangle^2}{\|c\|_2^2} \leq \|d\|_2^2$ and let $c_i = \sqrt{b_i}$, $d_i = a_i/\sqrt{b_i}$.

2.1 Distances Between Distributions

We use several similarity measures between (discrete) probability distributions X_0, X_1. Below we write $X_b(x)$ as a shorthand for $\Pr\{X_b = x\}$.

- Statistical Distance: $\Delta_{\mathsf{SD}}(X_0, X_1) = \frac{1}{2}\sum_x |X_0[x] - X_1[x]|$,
- (Squared) Hellinger Distance: $\Delta^2_{\mathsf{H}}(X_0, X_1) = \frac{1}{2}\sum_x (\sqrt{X_0[x]} - \sqrt{X_1[x]})^2$
- (Squared) Le Cam Distance: $\Delta^2_{\mathsf{LC}}(X_0, X_1) = \frac{1}{2}\sum_x \frac{(X_0[x]-X_1[x])^2}{X_0[x]+X_1[x]}$
- Renyi Divergence of order 1/2: $\Delta_{1/2}(X_0, X_1) = -2\ln \sum_x \sqrt{X_0[x]X_1[x]}$

It is well known that Δ_{SD}, Δ_{H} and Δ_{LC} are distance functions, i.e., they satisfy the triangle inequality. They are also closely related as follows.

Lemma 2. ([12, Section 7].). *For any two distributions X_0, X_1 we have*

$$\Delta^2_{\mathsf{H}}(X_0, X_1) \leq \Delta_{\mathsf{SD}}(X_0, X_1) \leq \sqrt{2}\Delta_{\mathsf{H}}(X_0, X_1)$$
$$\Delta_{\mathsf{H}}(X_0, X_1) \leq \Delta_{\mathsf{LC}}(X_0, X_1) \leq \sqrt{2}\Delta_{\mathsf{H}}(X_0, X_1).$$

In other words, Δ_{H} and Δ_{LC} are equivalent (up to a constant factor), while Δ_{SD} is polynomially related to them. As for the divergence $\Delta_{1/2}$, it easily follows from the definitions that it can be expressed as a monotonically increasing function of the Hellinger distance:

$$\Delta_{1/2}(X_0, X_1) = 2 \ln \frac{1}{1 - \Delta^2_{\mathsf{H}}(X_0, X_1)}.$$

Lemma 3. *For any two distributions X_0, X_1 such that $\Delta_{1/2}(X_0, X_1) < \infty$, we have*

$$\Delta^2_{\mathsf{H}}(X_0, X_1) \leq \frac{1}{2}\Delta_{1/2}(X_0, X_1) \leq \frac{\Delta^2_{\mathsf{H}}(X_0, X_1)}{1 - \Delta^2_{\mathsf{H}}(X_0, X_1)}.$$

In particular, if $\Delta^2_{\mathsf{H}}(X_0, X_1) \leq 1/2$, then $\Delta^2_{\mathsf{H}}(X_0, X_1) \leq \frac{1}{2}\Delta_{1/2}(X_0, X_1) \leq 2\Delta^2_{\mathsf{H}}(X_0, X_1)$.

Proof. Easily follows from the bounds $1 - (1/t) \leq \ln t \leq t - 1$ and relation $\Delta_{1/2}(X_0, X_1) = -2\ln(1 - \Delta^2_{\mathsf{H}}(X_0, X_1))$. See [15] for details.

2.2 Cryptographic Games

Cryptographic games are defined by one or more randomized, stateful programs $\mathcal{O}$ used by an adversary A to carry out an attack $A^{\mathcal{O}}$. When running $A^{\mathcal{O}}$, the adversary only has black-box access to $\mathcal{O}$, which is used as an oracle. There are two main categories of cryptographic games. In a *search* game, the final output of $A^{\mathcal{O}}$ is determined by $\mathcal{O}$ and indicates if the attack was successful. A *decision* game $\mathcal{O} = (\mathcal{O}_0, \mathcal{O}_1)$ is a pair of oracles with identical interfaces, so that an adversary A may interact with either of them $A^{\mathcal{O}_0}$, $A^{\mathcal{O}_1}$. This time it is A that produces an output at the end of the interaction. We refer to the set of all possible outputs of A as the *co-domain* of the adversary, and classify adversaries based on their co-domain. We consider three main classes of adversaries: traditional adversaries $A \in \mathcal{A}_{0,1}$ with co-domain $\{0, 1\}$, *aborting* adversaries $A \in \mathcal{A}_{\perp}$ with co-domain $\{0, 1, \perp\}$, and *fuzzy* adversaries $A \in \mathcal{A}_{\approx}$ with co-domain $[-1, 1]$. The goal of the adversary is to determine if it is interacting with either $\mathcal{O}_0$ or $\mathcal{O}_1$. A's advantage in distinguishing between $\mathcal{O}_0$ and $\mathcal{O}_1$ is defined later on. We write $A(\mathcal{O}_b)$ for the random variable describing the final output of A at the end of the interaction, and $A(\mathcal{O})$ as an abbreviation for the pair of output distributions $(A(\mathcal{O}_0), A(\mathcal{O}_1))$ over the co-domain of A. We remark that the output distribution $A(\mathcal{O}_b)$ is defined over the internal randomness of both A and $\mathcal{O}_b$. We write $A(\mathcal{O}_b; r)$ when we want to make the randomness of A explicit.

In the simplest, prototypical example $\mathcal{O} = (\mathcal{O}_0, \mathcal{O}_1)$ is a pair of probability distributions over a common set Ω. The only interaction between A and $\mathcal{O}_b$ is to receive a single sample $x \leftarrow \mathcal{O}_b$ from Ω, upon which $A(x)$ produces its final output. In other words, the adversary A is just a (possibly randomized) algorithm with input in Ω. For simplicity, the reader may keep this simple case in mind throughout the paper, instead of arbitraty games. In any case, once a game $\mathcal{O}$ and adversary A have been chosen, the output $A(\mathcal{O}) = (A(\mathcal{O}_0), A(\mathcal{O}_1))$ is always a pair of probability distributions.

Cryptographic protocols can be parameterized by other cryptographic primitives or distributions used as building blocks. So, for example, we may write P^Y for a cryptographic program that uses a probability distribution Y, and $P^{Y'}$ for the same program run with a different distribution Y'. Similarly, security games $(\mathcal{D}_0^Y, \mathcal{D}_1^Y)$ can be parameterized by Y.

We remark that the running time of an adversary A against a game $\mathcal{D}$ does not include the time required to run $\mathcal{D}$ in the interaction $A(\mathcal{D})$. In other words, we only account for the time taken by A to write its oracle queries and read the answers. We consider adversaries running in strict (e.g. polynomial) time, i.e., we assume that the running time of A in a run $A(\mathcal{D}_b)$ does not depend on how $\mathcal{D}_b$ answers the oracle queries. In particular, A has the same running time in $A(\mathcal{D}_0)$ and $A(\mathcal{D}_1)$. The running time of an adversary A is denoted by T_A or $T(A)$.

In some settings it is useful to define also a notion of running time for the game $\mathcal{D}$. However, it should be clear that the (total) running time of $\mathcal{D}$ in an execution $A(\mathcal{D})$ typically depends on the adversary A.[8] The time taken to run $\mathcal{D}$ in an execution $A(\mathcal{D})$ is denoted $T_{\mathcal{D}}^A$. Then, we can define the running time of a game $\mathcal{D}$ relative to the running time of A as follows.

Definition 1. *The (relative) running time of* $\mathcal{D}$ *is defined as the maximum*

$$T_{\mathcal{D}} = \sup_A \frac{T_{\mathcal{D}}^A}{T_A}$$

over all possible adversaries A*.*

For decision games $(\mathcal{D}_0, \mathcal{D}_1)$ we always assume that $\mathcal{D}_0$ and $\mathcal{D}_1$ have the same running time. Using this definition, the total running time to run $A(\mathcal{D})$ (including both the time for A and for $\mathcal{D}$) can be bounded as

$$T_{A(\mathcal{D})} = T_A + T_{\mathcal{D}}^A \leq T_A \cdot (1 + T_{\mathcal{D}}) \,.$$

In the asymptotic setting both the game $\mathcal{D}_\kappa = (\mathcal{D}_{\kappa,0}, \mathcal{D}_{\kappa,1})$ and adversary A_κ are parametrized by a security parameter κ, and all quantities (e.g., A_κ's advantage in winning a decision game, its running time T_{A_κ}, etc.) become functions of κ. Notice that A_κ may depend arbitrarily on κ, i.e., we consider non-uniform adversaries. Technically, $A(\kappa, \tau_\kappa)$ is an algorithm that takes as input the security parameter κ and an advice string τ_κ that depends on the security parameter. However, we will make only very limited use of non-uniformity: in most of our results τ_κ is a very short (typically constant size, independently of κ) string. So, the non-uniformity can be easily eliminated by running $A(\kappa, \tau)$ on all possible value of τ, estimating A's advantage, and then picking the best value of τ to carry out the attack.

[8] This is most obvious when $\mathcal{D}$ is a game where A may issue an arbitrary number of calls to the game oracles.

2.3 Bit Security

Consider an adversary A against a decision game $\eth = (\eth_0, \eth_1)$, where $A(\eth_b)$ may output $0, 1$ or some other values. Throughout the paper we will use the following definitions and notation:

$$\begin{aligned} \text{(success probability)} \quad & \beta_A = \Pr[A(\eth_b) = b] \\ \text{(failure probability)} \quad & \bar{\beta}_A = \Pr[A(\eth_b) = 1 - b] \\ \text{(output probability)} \quad & \alpha_A = \beta_A + \bar{\beta}_A \\ \text{(distinguishing gap)} \quad & \delta_A = \beta_A - \bar{\beta}_A \end{aligned}$$

where all probabilities are computed over the random choice of $b \leftarrow \{0,1\}$, and the randomness of $\eth_b$ and A. Notice that α_A equals the probability that the output of A is in $\{0,1\}$. So, for standard adversaries $A \in \mathcal{A}_{0,1}$ that always output a bit $A(\eth_b) \in \{0,1\}$, we have $\alpha_A = 1$ and $\delta_A = 2\beta_A - 1 = \Pr\{A(\eth_1) = 1\} - \Pr\{A(\eth_0) = 1\}$. But we will use the definition of $\beta_A, \bar{\beta}_A, \alpha_A$ and δ_A also for unrestricted adversaries that may output values outside of $\{0,1\}$. It is well-known that, in the case of probability distributions $\mathcal{X} = (X_0, X_1)$, the highest possible distinguishing gap equals the statistical distance $\Delta_{\mathsf{SD}}(X_0, X_1) = \max_{A \in \mathcal{A}_{0,1}} \delta_A$ and it is achieved by a very simple adversary

$$A^{\mathcal{X}}_{\mathsf{SD}}(x) = \begin{cases} 0 \text{ if } \Pr[X_0 = x] > \Pr[X_1 = x] \\ 1 \text{ if } \Pr[X_0 = x] < \Pr[X_1 = x] \end{cases} \tag{2}$$

(When $\Pr[X_0 = x] = \Pr[X_1 = x]$, the output of A can be chosen arbitrarily without affecting the gap δ.) This is easily generalized to arbitrary decision games $\eth = (\eth_0, \eth_1)$, where

$$\Delta_{\mathsf{SD}}(\eth_0, \eth_1) = \max_{A \in \mathcal{A}_{0,1}} \delta_A. \tag{3}$$

Since $\delta_A = \beta_A - \bar{\beta}_A$ is the difference between two probabilities, and the maximum over all A is non-negative, we always have $\Delta_{\mathsf{SD}}(\eth_0, \eth_1) \in [0,1]$.

The MW Bit Security Measure. Micciancio and Walter [10] suggested to use a more general class of adversaries $\mathcal{A}_\perp$, which output either 0, 1, or a special "don't know" symbol $\perp$, and demonstrated that these adversaries, together with an appropriate notion of advantage, allow to resolve several theoretical paradoxes related to the definition of a cryptographically meaningful notion of "bit security". (The reader is referred to [10] for intuition and justification of this definition.)

Definition 2 (MW Advantage). *For any (possibly randomized) MW distinguisher $A \in \mathcal{A}_\perp$ and decision game $\eth = (\eth_0, \eth_1)$, the advantage of A is*[9]

$$\mathsf{adv}^{\mathsf{MW}}_{\eth}(A) = \frac{\delta_A^2}{\alpha_A} = \frac{(\beta_A - \bar{\beta}_A)^2}{\beta_A + \bar{\beta}_A}$$

[9] This is syntactically different, but perfectly equivalent to the definition given in [10], which defines the advantage as $\alpha_A \cdot (2\beta_A^* - 1)^2$, where $\beta_A^* = \beta_A / \alpha_A$.

The (squared) MW distance between two distributions is

$$\Delta^2_{\mathsf{MW}}(\mathcal{D}_0, \mathcal{D}_1) = \sup_{A \in \mathcal{A}_\perp} \mathsf{adv}^{\mathsf{MW}}_{\mathcal{D}}(A) \in [0,1]. \tag{4}$$

If we restrict our attention to "non-aborting" adversaries $A \in \mathcal{A}_{0,1}$, we have $\alpha_A = 1$, and $\mathsf{adv}^{\mathsf{MW}}_{\mathcal{D}}(A) = \delta_A^2$ is the square of the distinguishing gap. This immediately gives the following inequality.

Lemma 4. *For any decision game $\mathcal{D} = (\mathcal{D}_0, \mathcal{D}_1)$, we have*

$$\Delta_{\mathsf{SD}}(\mathcal{D}_0, \mathcal{D}_1) \le \Delta_{\mathsf{MW}}(\mathcal{D}_0, \mathcal{D}_1).$$

Proof. Using (3) and the definition of Δ_{MW}, we get

$$\Delta_{\mathsf{SD}}(\mathcal{D}_0, \mathcal{D}_1) = \sup_{A \in \mathcal{A}_{0,1}} \delta_A = \sup_{A \in \mathcal{A}_{0,1}} \sqrt{\mathsf{adv}^{\mathsf{MW}}_{\mathcal{D}}(A)} \le \Delta_{\mathsf{MW}}(\mathcal{D}_0, \mathcal{D}_1)$$

where the inequality follows from taking the supremum over a larger set $A \in \mathcal{A}_\perp$.

It is also easy to see that the MW distance satisfies the data processing inequality.

Lemma 5 (Data-Processing Inequality). *For any decision game $\mathcal{D} = (\mathcal{D}_0, \mathcal{D}_1)$ and game transformation[10] Γ, we have that*

$$\Delta_{\mathsf{MW}}(\Gamma(\mathcal{D}_0), \Gamma(\mathcal{D}_1)) \le \Delta_{\mathsf{MW}}(\mathcal{D}_0, \mathcal{D}_1).$$

Proof. For any aborting adversary A, define $A^\Gamma(\mathcal{D}_b) := A(\Gamma(\mathcal{D}_b))$. It is straightforward to see that

$$\Delta^2_{\mathsf{MW}}(\Gamma(\mathcal{D}_0), \Gamma(\mathcal{D}_1)) = \sup_{A^\Gamma} \mathsf{adv}^{\mathsf{MW}}_{\mathcal{D}}(A^\Gamma) \le \sup_{A} \mathsf{adv}^{\mathsf{MW}}_{\mathcal{D}}(A) = \Delta^2_{\mathsf{MW}}(\mathcal{D}_0, \mathcal{D}_1).$$

We will use the following construction from [10] to transform an aborting adversary $A \in \mathcal{A}_\perp$ to one with only two possible output values. For any adversary $A \in \mathcal{A}_\perp$, decision game $\mathcal{D} = (\mathcal{D}_0, \mathcal{D}_1)$, and value $z \in \{0, 1, \perp\}$, let $\hat{b} = A^{A(\mathcal{D})}_{\mathsf{SD}}(z) \in \{0,1\}$ be the bit such that $\Pr\{A(\mathcal{D}_{\hat{b}}) = z\}$ is highest. (This bit can be given as a non-uniform advice, or estimated probabilistically by repeatedly running $A(\mathcal{D}_{\hat{b}})$ for $\hat{b} \in \{0,1\}$ with independent randomness.) Given $\hat{b}$, let

$$A^z(\mathcal{D}_b) = \text{if } (A(\mathcal{D}_b) = z) \text{ then } \hat{b} \text{ else } \perp \tag{5}$$

be the modified adversary that first runs $z' \leftarrow A(\mathcal{D}_b)$, and then outputs $\hat{b}$ if $z' = z$ or $\perp$ otherwise. Notice that A^z differs from A just by a relabeling of its output. So, it has the same running time $T(A^z) = T(A)$.

Lemma 6 ([10, Lemma 1]). *For any pair decision game $\mathcal{D} = (\mathcal{D}_0, \mathcal{D}_1)$, aborting adversary $A \in \mathcal{A}_\perp$, and value $z \in \{0, 1, \perp\}$, the modified adversary A^z in (5) has advantage*

$$\mathsf{adv}^{\mathsf{MW}}_{\mathcal{D}}(A^z) = \frac{(\Pr\{A(\mathcal{D}_0) = z\} - \Pr\{A(\mathcal{D}_1) = z\})^2}{2(\Pr\{A(\mathcal{D}_0) = z\} + \Pr\{A(\mathcal{D}_1) = z\})}.$$

[10] A transformation is simply a game $\Gamma(\mathcal{D}_b)$ with oracle access to $\mathcal{D}_b$. Applying Γ to a game $\mathcal{D}$ defines a new game $(\Gamma(\mathcal{D}_0), \Gamma(\mathcal{D}_1))$.

The WY Bit Security Measure. In [14], an alternative bit security measure was introduced. The definition is parameterized by a "high enough" probability threshold $\mu \approx 1$, but it can be shown that the precise value of μ has only a marginal impact on the definition. An equivalent quantity (without the parameter μ) is also defined in terms of the Renyi divergence of order 1/2.

Definition 3. *Let $\eth = (\eth_0, \eth_1)$ be a decision game, $\mu \in [0,1]$, and $\epsilon_{A,B_k} := \Pr_b[B_k(A(\eth_b)^k) = b]$, where $A \in \mathcal{A}_{0,1}$, $k \in \mathbb{N}$, $B_k : \{0,1\}^k \to \{0,1\}$, and X^k is the product distribution over $\{0,1\}^k$ of k independent copies of $X = A(\eth_b)$. Define*

$$\mathsf{WY}^{\mu}_{\eth}(A) = \min_k \min_{B_k} \{\log_2(k \cdot T_A) \mid \epsilon_{A,B_k} \geq 1 - \mu\}, \qquad \mathsf{WY}^{\mu}_{\eth} := \min_{A \in \mathcal{A}_{0,1}} \mathsf{WY}^{\mu}_{\eth}(A). \tag{6}$$

$$\mathsf{WY}_{\eth}(A) := \log_2 T(A) + \log_2 \left\lceil \frac{1}{\Delta_{1/2}(A(\eth_0), A(\eth_1))} \right\rceil, \qquad \mathsf{WY}_{\eth} := \min_{A \in \mathcal{A}_{0,1}} \mathsf{WY}_{\eth}(A). \tag{7}$$

We say that two bit security measures are equivalent if they differ by an additive constant factor. While not highlighted as a formal statement, [14] shows that all these measures are essentially equivalent.

Lemma 7 ([14, implicit]). *For any distinguishing game $\eth := (\eth_0, \eth_1)$, for any constants $\mu \leq \mu'$, one has that*

$$\left| \mathsf{WY}^{\mu}_{\eth} - \mathsf{WY}^{\mu'}_{\eth} \right| \leq O(1). \tag{8}$$

$$|\mathsf{WY}_{\eth} - \mathsf{WY}^{\mu}_{\eth}| \leq O(1), \tag{9}$$

Proof. The (stronger) bound

$$\forall A \in \mathcal{A}_{0,1} : \left| \mathsf{WY}^{\mu}_{\eth}(A) - \mathsf{WY}^{\mu'}_{\eth}(A) \right| \leq \ln(\ln(\frac{1}{4\mu^2})) \leq O(1) \tag{10}$$

follows from simple algebraic manipulations of [14, Lemmas 4 and 6], which bound the minimum k in (6) via

$$\frac{\ln(\frac{1}{4\mu})}{\Delta_{1/2}(A(\eth_0), A(\eth_1))} \leq k \leq \left\lceil \frac{\ln(\frac{1}{4\mu^2})}{\Delta_{1/2}(A(\eth_0), A(\eth_1))} \right\rceil. \tag{11}$$

Multiplying by T_A and taking logarithms yields nearly matching upper and lower bounds on $\mathsf{WY}^{\mu}_{\eth}$, which suffice to establish (10). One then gets the claimed result by minimizing (10) over A.

Equivalence with the MW bit security is proved in [15], but technically only for aborting adversaries, which we denote $\mathsf{WY}^{\perp}_{\eth} = \min_{A \in \mathcal{A}_{\perp}} \mathsf{WY}_{\eth}(A)$.

Lemma 8 ([15, Theorems 1 and 2]). *For any distinguishing game $\eth := (\eth_0, \eth_1)$,*

$$\left| \mathsf{WY}^{\perp}_{\eth} - \mathsf{MW}_{\eth} \right| \leq O(1).$$

Note that the measure $\mathsf{WY}^{\perp}_{\eth}$ is not *a priori* equal to $\mathsf{WY}_{\eth}$, as minimizing over a larger set $\mathcal{A}_{\perp}$ may produce smaller values. So, Lemma 8 does not imply that $\mathsf{WY}_{\eth}$ and $\mathsf{MW}_{\eth}$ are equivalent. Still, this is true, as we will show in Sect. 4.

Computational/Statistical Bit Security. Sometimes, in cryptography, one can achieve a strong notion of security, where no adversary can break a cryptographic function with high probability, regardless of the computational cost incurred by the attack. In the MW bit-security framework, the number of bits of statistical security of a decision game $\eth$ can be defined as follows.

Definition 4. *A distinguishing game $\eth = (\eth_0, \eth_1)$ has s bits of statistical security if for every adversary A, $\mathsf{adv}^{\mathsf{MW}}_{\eth}(A) \leq 2^{-s}$.*

Contrast this with the definition of (computational) bit-security, where the requirement is that $\mathsf{adv}^{\mathsf{MW}}_{\eth}(A) \leq T(A) \cdot 2^{-c}$. It immediately follows from the definition that any problem achieving s bits of statistical security, also offers s bits of computational security. So, statistical bit-security is a strengthening of computational bit-security. In particular, when combining computational and statistical primitives within a single protocols, one can treat all of them has achieving a given number $c = s$ of computational security bits. However, this is often undesirable in practice because one typically wants to use a higher value of c than for s. In order to combine computational and statistical bit-security analysis in an efficient manner, [8] proposes the following notion of *computational-statistical* bit-security.

Definition 5 ([8]). *A distinguishing game $\eth$ is said to have (c, s)-bits of security if for any adversary $A \in \mathcal{A}_{\perp}$,*

$$\mathsf{adv}^{\mathsf{MW}}_{\eth}(A) \leq \max(T(A)2^{-c}, 2^{-s}),$$

i.e., either $c \leq \log_2 \frac{T(A)}{\mathsf{adv}^{\mathsf{MW}}_{\eth}(A)}$, or $s \leq \log_2 \frac{1}{\mathsf{adv}^{\mathsf{MW}}_{\eth}(A)}$.

The notions of computational and statistical security corresponds to the following special cases of (c, s)-security:

- A problem has c bits of computational security iff it is (c, ∞)-bit secure
- A problem has s bits of computational security iff if is (∞, s)-bit secure.

Since any problem offering s bits of statistical security also offers s bits of computational security, (c, s)-bit security is equivalent to $(\max(c, s), s)$-bit security. In other words, one can always assume $c \geq s$. In particular, computational security can be equivalently formulated as (c, c)-bit security, rather than (c, ∞).

(c, s)-security is easily defined for search problems as well: A search game $\eth$ has (c, s)-bits of security if any adversary A has success probability[11] $\Pr\{A(\eth)\}$ at most $\max(T(A)2^{-c}, 2^{-s})$.

3 Structure and Properties of Optimal MW Adversaries

In this section we characterize the MW adversaries achieving optimal advantage, and prove some useful properties about them. This is done by introducing an alternative, more general, class of adversaries (which we call "fuzzy" adversaries,) that still achieves the same optimal advantage (and bit security) of standard MW adversaries. We use the added flexibility provided by fuzzy adversaries to investigate optimal adversarial strategies.

MW adversaries are generalized as follows. Recall that the output of an MW distinguisher is either a bit $b \in \{0, 1\}$, representing a *high confidence* decision between the two distributions, or a special symbol $\perp$ expressing *no confidence*. We generalize this to distinguishers for which the output confidence level can vary continuously from 0 (no confidence) to 1 (high confidence). For this type of distinguishers, it is convenient to map the two values $b \in \{0, 1\}$ to a sign

$$\tilde{b} = (-1)^b = (1 - 2b) = \pm 1 \tag{12}$$

so that the output of A can be described by a single number $\sigma \in [-1, 1]$, with $\mathsf{sign}(\sigma) = \sigma/|\sigma| = \tilde{b} \in \{\pm 1\}$ representing the decision bit and $|\sigma| \in [0, 1]$ the confidence level.[12] We also set $\tilde{\perp} = 0$, so that any MW distinguisher A with output $A(\eth_b) = y \in \{0, 1, \perp\}$ can be represented by a fuzzy one $\tilde{A}$ with output $\tilde{A}(\eth_b) = \tilde{y} \in \{1, -1, 0\} \subset [-1, 1]$. Notice that this transformation preserves the cost of the adversary $T(\tilde{A}) = T(A)$ as the only difference between the two is the symbol used to encode the final output. We write $\tilde{\mathcal{A}_\perp} = \{\tilde{A} \mid A \in \mathcal{A}_\perp\}$ for the set of aborting adversaries with this alternative output representation.

Definition 6 (Fuzzy Distinguisher). *A fuzzy distinguisher for a decision game $\eth = (\eth_0, \eth_1)$ is a (possibly randomized) adversary A with output in $[-1, 1]$. The* advantage *and* bit-security *of A in the game $\eth$ are $\mathsf{adv}^{\mathsf{MW}}_{\eth}(A) = \frac{\tilde{\delta}_A^2}{\tilde{\alpha}_A}$ and $\mathsf{MW}_{\eth}(A) = \log_2(T(A)/\mathsf{adv}^{\mathsf{MW}}_{\eth}(A))$ where*

$$\tilde{\delta}_A = \mathbb{E}_b[\tilde{b} \cdot A(\eth_b)] \qquad \text{and} \qquad \tilde{\alpha}_A := \mathbb{E}_b\left[|A(\eth_b)|\right]$$

are the correlation (between the correct result and the output of A) and expected confidence of A. The set of all possible fuzzy distinguishers is denoted $\mathcal{A}_{\approx}$.

Note that $\tilde{\mathcal{A}_\perp} \subset \mathcal{A}_{\approx}$, so we can view aborting adversaries as a special case of fuzzy adversaries. The following lemma shows that the definition of advantage

[11] We recall that for a search problem, the output of $A(\eth)$ is determined by the game $\eth$.

[12] When $\sigma = 0$, the confidence $|\sigma| = 0$ is zero, and the decision $\mathsf{sign}(\sigma)$ is irrelevant. For concreteness, we define $\mathsf{sign}(0) = 0$.

given in Definition 6 respects this identification. This justifies the use of the same notation $\mathsf{adv}^{\mathsf{MW}}_{\eth}(A)$ and $\mathsf{MW}_{\eth}(A)$ for the advantage and bit security of both aborting $A \in \mathcal{A}_{\perp}$ and fuzzy adversaries $A \in \mathcal{A}_{\approx}$.

Lemma 9. *For any aborting adversary $A \in \mathcal{A}_{\perp}$ and corresponding fuzzy adversary $\tilde{A} \in \mathcal{A}_{\approx}$ we have $\tilde{\delta}_{\tilde{A}} = \delta_A$, $\tilde{\alpha}_{\tilde{A}} = \alpha_A$, $T(\tilde{A}) = T(A)$, $\mathsf{adv}^{\mathsf{MW}}_{\eth}(\tilde{A}) = \mathsf{adv}^{\mathsf{MW}}_{\eth}(A)$ and $\mathsf{MW}_{\eth}(\tilde{A}) = \mathsf{MW}_{\eth}(A)$.*

Proof. It is easy to check that $\tilde{delta}_{\tilde{A}} = \delta_A$ and $\tilde{\alpha}_{\tilde{A}} = \alpha_A$ by evaluating the expectations over the set $0, 1, -1$ of all possible values. It follows that $\mathsf{adv}^{\mathsf{MW}}_{\eth}(\tilde{A}) = \tilde{\delta}^2_{\tilde{A}}/\tilde{\alpha}_{\tilde{A}} = \delta^2_A/\alpha_A = \mathsf{adv}^{\mathsf{MW}}_{\eth}(A)$. Finally, A and $\tilde{A}$ have the same running time $T(A) = T(\tilde{A})$. So, we also have

$$\mathsf{MW}_{\eth}(\tilde{A}) = \log_2(T(\tilde{A})/\mathsf{adv}^{\mathsf{MW}}_{\eth}(\tilde{A})) = \log_2(T(A)/\mathsf{adv}^{\mathsf{MW}}_{\eth}(A)) = \mathsf{MW}_{\eth}(A).$$

3.1 Equivalence of Aborting and Fuzzy Adversaries

Using fuzzy adversaries, we may define the maximum (statistical) advantage in attacking a decision game $\eth$ as

$$(\Delta^{\approx}_{\mathsf{MW}}(\eth))^2 = \sup\{\mathsf{adv}^{\mathsf{MW}}_{\eth}(A) \mid A \in \mathcal{A}_{\approx}\},$$

and similarly for bit security

$$\mathsf{MW}_{\approx}(\eth) = \inf\{\mathsf{MW}_{\eth}(A) \mid A \in \mathcal{A}_{\approx}\}.$$

Since we are optimizing over a larger class of adversaries $\mathcal{A}_{\approx} \supset \mathcal{A}_{\perp}$, it immediately follows from the definitions that $\Delta_{\mathsf{MW}}(\eth) \leq \Delta^{\approx}_{\mathsf{MW}}(\eth)$ and $\mathsf{MW}(\eth) \geq \mathsf{MW}_{\approx}(\eth)$, and in principle these inequalities could be strict. But, as we will see, this is not the case, i.e., aborting and fuzzy adversaries define precisely the same notion of advantage and bit security for decision games. This is proved using the following transformation.

Lemma 10. *Let $\mathsf{N}\colon \mathcal{A}_{\approx} \to \mathcal{A}_{\perp}$ be the transformation*[13]

$$\mathsf{N}[A](\eth_b; r) = \begin{cases} \frac{1-\mathsf{sign}(A(\eth_b;r))}{2} & \text{with probability } |A(\eth_b; r)| \\ \perp & \text{otherwise.} \end{cases}$$

Then, for any decision game $\eth = (\eth_0, \eth_1)$ and adversary $A \in \mathcal{A}_{\approx}$, we have

$$\mathsf{adv}^{\mathsf{MW}}_{\eth}(A) = \mathsf{adv}^{\mathsf{MW}}_{\eth}(\mathsf{N}[A]).$$

[13] More precisely, $\mathsf{N}[A]$ is the aborting adversary that runs the fuzzy attack $a \leftarrow A(\eth_b) \in [-1, 1]$, and then outputs $(1 - \mathsf{sign}(a))/2$ with probability $|a|$ and $\perp$ with probability $1-|a|$. Note that the output of $\mathsf{N}[A]$ is always in $\{0, 1, \perp\}$, i.e., $\mathsf{N}[A] \in \mathcal{A}_{\perp}$ is a valid aborting adversary.

Proof. We have that

$$\begin{aligned}
\delta_{\mathsf{N}[A]} &= \Pr_{b,r}[\mathsf{N}[A](\eth_b; r) = b] - \Pr_{b,r}[\mathsf{N}[A](\eth_b; r) = 1 - b] \\
&= \mathbb{E}_b \left[|A(\eth_b)| \cdot \Pr \left[\frac{1 - \mathsf{sign}(A(\eth_b))}{2} = b \right] \right. \\
&\qquad \left. - |A(\eth_b)| \cdot \Pr \left[\frac{1 - \mathsf{sign}(A(\eth_b))}{2} = 1 - b \right] \right] \\
&= \mathbb{E}_b[|A(\eth_b)| \cdot (\Pr[\mathsf{sign}(A(\eth_b)) = 1 - 2b] - \Pr[\mathsf{sign}(A(\eth_b)) = -(1 - 2b)])] \\
&= \mathbb{E}_b[|A(\eth_b)| \cdot (\Pr[\mathsf{sign}(A(\eth_b)) = (-1)^b] - \Pr[\mathsf{sign}(A(\eth_b)) = -(-1)^b])] \\
&= \mathbb{E}_b[|A(\eth_b)| \cdot (\Pr[(-1)^b \cdot \mathsf{sign}(A(\eth_b)) = 1] - \Pr[(-1)^b \cdot \mathsf{sign}(A(\eth_b)) = -1])] \\
&= \mathbb{E}_b[|A(\eth_b)| \cdot \mathbb{E}[(-1)^b \cdot \mathsf{sign}(A(\eth_b))]] \\
&= \mathbb{E}_b \left[(-1)^b \cdot A(\eth_b) \right] = \delta_A,
\end{aligned}$$

and

$$\alpha_{\mathsf{N}[A]} = \Pr_{b,r}[\mathsf{N}[A](\eth_b; r) \neq \perp] = \mathbb{E}_{b,r}[|A(\eth_b; r)|] = \alpha_A.$$

It then follows that $\mathsf{adv}^{\mathsf{MW}}_{\eth}(A) = \frac{{\delta_A}^2}{\alpha_A} = \frac{{\delta_{\mathsf{N}[A]}}^2}{\alpha_{\mathsf{N}[A]}} = \mathsf{adv}^{\mathsf{MW}}_{\eth}(\mathsf{N}[A])$, i.e. N preserves the advantage.

Clearly, the transformation N also preserves the complexity of the adversary $T(\mathsf{N}[A]) \approx T[A]$, as the additional operations performed by $\mathsf{N}[A]$ have negligible cost. It immediately follows that aborting and fuzzy adversaries are equivalent, both for statistical and computational bit security.

Theorem 1. *Aborting and Fuzzy MW adversaries are equivalent, i.e., they define the same notions of advantage and bit security*

$$\begin{aligned}
\Delta^{\approx}_{\mathsf{MW}}(\eth) &= \Delta_{\mathsf{MW}}(\eth) \\
\mathsf{MW}_{\approx}(\eth) &= \mathsf{MW}(\eth).
\end{aligned}$$

Proof. We need to show that $\Delta_{\mathsf{MW}}(\eth) \geq \Delta^{\approx}_{\mathsf{MW}}(\eth)$ and $\mathsf{MW}(\eth) \leq \mathsf{MW}_{\approx}(\eth)$. For any $A \in \mathcal{A}_{\approx}$, the aborting adversary $\mathsf{N}[A] \in \mathcal{A}_{\perp}$ satisfies

$$\mathsf{adv}^{\mathsf{MW}}_{\eth}(A) = \mathsf{adv}^{\mathsf{MW}}_{\eth}(\mathsf{N}[A]) \leq \sup_{A'} \mathsf{adv}^{\mathsf{MW}}_{\eth}(A') = \Delta^2_{\mathsf{MW}}(\eth_0, \eth_1).$$

Therefore, $(\Delta^{\approx}_{\mathsf{MW}}(\eth))^2 = \sup_A \mathsf{adv}^{\mathsf{MW}}_{\eth}(A) \leq \Delta^2_{\mathsf{MW}}(\eth)$. A similar argument works for bit security, using the fact that $T(A) \approx T(\mathsf{N}(A))$.

3.2 Convexity and Determinism

In general, cryptographic adversaries can use randomness. Using *fuzzy* adversaries it is easy to turn any randomized adversary into a deterministic one. In the following lemma we give a simple transformation from (randomized) aborting adversaries to deterministic fuzzy ones. For simplicity, we present the lemma

for the simple problem of distinguishing between two probability distributions $\mathcal{X} = (X_0, X_1)$. A more general statement for arbitrary games will be proved later in this section.

Lemma 11. *Let* $\mathsf{F}\colon \mathcal{A}_\perp \to \mathcal{A}_\approx$ *be the transformation*

$$\mathsf{F}[A](x) = \Pr_r[A(x;r) = 0] - \Pr_r[A(x;r) = 1]$$

mapping a (randomized) aborting adversary A *to a deterministic fuzzy adversary* $\mathsf{F}[A] \in \mathcal{A}_\approx$*. Then, for any decision problem* $\mathcal{X} = (X_0, X_1)$ *and adversary* $A \in \mathcal{A}_\perp$ *we have*

$$\mathsf{adv}^{\mathsf{MW}}_{\ominus}(A) \le \mathsf{adv}^{\mathsf{MW}}_{\mathcal{X}}(\mathsf{F}[A]).$$

In particular, the optimal advantage $\Delta^2_{\mathsf{MW}}(\ominus)$ *is achieved by a* deterministic $A \in \mathcal{A}_\approx$*.*

Proof. We first show that $\delta_{\mathsf{F}[A]} = \delta_A$. We have that

$$\begin{aligned}
\delta_{\mathsf{F}[A]} &= \mathbb{E}_b[\tilde{b} \cdot \mathsf{F}[A](X_b)] \\
&= \mathbb{E}_b[(-1)^b \cdot (\Pr_r[A(X_b;r) = 0] - \Pr_r[A(X_b;r) = 1])] \\
&= \frac{1}{2}\left(\mathbb{E}[(-1) \cdot (\Pr_r[A(X_1;r) = 0] - \Pr_r[A(X_1;r) = 1])]\right) \\
&+ \frac{1}{2}\left(\mathbb{E}[(+1) \cdot (\Pr_r[A(X_0;r) = 0] - \Pr_r[A(X_0;r) = 1])]\right) \\
&= \frac{\mathbb{E}[\Pr_r[A(X_1;r) = 1]] + \mathbb{E}[\Pr_r[A(X_0;r) = 0]]}{2} \\
&- \frac{\mathbb{E}[\Pr_r[A(X_1;r) = 0]] + \mathbb{E}[\Pr_r[A(X_0;r) = 1]]}{2} \\
&= \mathbb{E}_b[\Pr_r[A(X_b;r) = b]] - \mathbb{E}_b[\Pr_r[A(X_b;r) = 1 - b]] \\
&= \beta_A - \bar{\beta}_A = \delta_A.
\end{aligned}$$

We next show that $\alpha_{\mathsf{F}[A]} \le \alpha_A$. We have that

$$\begin{aligned}
\alpha_{\mathsf{F}[A]} &= \mathbb{E}_b[|\mathsf{F}[A](X_b)|] \\
&= \mathbb{E}_b\left[\left|\Pr_r[A(X_b;r) = 0] - \Pr_r[A(X_b;r) = 1]\right|\right] \\
&\le \mathbb{E}_b\left[\Pr_r[A(X_b;r) = 0] + \Pr_r[A(X_b;r) = 1]\right] = \alpha_A.
\end{aligned}$$

It follows that $\mathsf{adv}^{\mathsf{MW}}_{\mathcal{X}}(A) = \frac{{\delta_A}^2}{\alpha_A} \le \frac{{\delta_{\mathsf{F}[A]}}^2}{\alpha_{\mathsf{F}[A]}} = \mathsf{adv}^{\mathsf{MW}}_{\mathcal{X}}(\mathsf{F}[A])$.

Notice that the result of the transformation $\mathsf{F}[A]$ is not in general an efficient algorithm, because it requires the computation of the probabilities[14]

[14] Naturally, one could approximate these probabilities in a relatively efficient manner by repeatedly running $A(x;r_i)$ on a given input x and many independent random r_i. However, this would result in a randomized algorithm.

$\Pr_r[A(x;r) = b]$ for $b = 0, 1$. So, Lemma 11 says little about the (computational) bit security under deterministic attacks. Moreover, it says nothing about the existence of deterministic *aborting* adversaries $A \in \mathcal{A}_\perp$ because $\mathsf{F}[A]$ is fuzzy.[15]

We would like to prove a similar result (for arbitrary decision games) that produces deterministic aborting adversaries, and address the efficiency issue (at least in the non-uniform setting). We will show that for any randomized aborting adversary A there is a value of the randomness r such that the deterministic adversary $A_r(\cdot) = A(\cdot; r)$ is at least as good as A. But before doing so, we observe that (perhaps contrary to intuition) this is not generally true for arbitrary notions of advantage.

Lemma 12. *There is a pair of efficiently samplable distributions $\mathcal{X} = (X_0, X_1)$, randomized distinguisher $A(x;r)$ and advantage function $\mathsf{adv}^\star_{\mathcal{X}}$ such that $\mathsf{adv}^\star_{\mathcal{X}}(A)$ is strictly bigger than $\mathsf{adv}^\star_{\mathcal{X}}(A(\cdot;r))$ for all r.*

Proof. Let X_0 and X_1 be the uniform distributions over $\{0\}$ and $\{0, 1, 2, 3\}$ respectively, and consider a randomized distinguisher $A(x;r)$ using a single bit of randomness $r \in \{0, 1\}$ that works as follows: if $x \le 2r$ then $A(x;r) = 0$, and $A(x;r) = \perp$ otherwise.

Consider the output of $A(x;r)$ when $x \leftarrow X_b$ for $b \leftarrow \{0, 1\}$. It is easy to see that $A(\cdot;r)$ is correct precisely when $b = 0$. So, A has success probability $\beta = 1/2$ regardless of the value of the randomness r. On the other hand, the failure probability is $\bar{\beta}_0 = 1/8$ when $r = 0$ (for $b = 1$ and $x = 0$), $\bar{\beta}_1 = 3/8$ when $r = 1$ (for $b = 1$ and $x \in \{0, 1, 2\}$) and $\bar{\beta} = (\bar{\beta}_0 + \bar{\beta}_1)/2 = 1/4$ when r is chosen at random. Now define the advantage function[16]

$$\mathsf{adv}^\star_{\mathcal{X}}(A) = \left|\beta_A - \sin(2\pi\bar{\beta}_A)\right| \cdot (\beta_A - \bar{\beta}_A).$$

Using this function we can compute $\mathsf{adv}^\star_{\mathcal{X}}(A) \approx 0.125$, $\mathsf{adv}^\star_{\mathcal{X}}(A(\cdot;0)) \approx 0.077$ and $\mathsf{adv}^\star_{\mathcal{X}}(A(\cdot;1)) \approx 0.025$. So, the advantage of the randomized adversary A is strictly bigger than both $A(\cdot;0)$ and $A(\cdot;1)$.

We prove the existence of deterministic optimal aborting adversaries using a convexity argument. For any adversaries $A, B \in \mathcal{A}_\perp$ and $\theta \in [0, 1]$, define the convex combination $C = \theta \cdot A + (1 - \theta) \cdot B$ as the (randomized) adversary that runs A with probability θ and B with probability $1 - \theta$. Notice that the convex combination is taken over the randomness, not the output of the adversaries, so that the result is still an aborting adversary in $\mathcal{A}_\perp$.

[15] Note that turning $\mathsf{F}[A]$ into an aborting adversary $\mathsf{N}[\mathsf{F}[A]]$ using Lemma 10 does not work, because the result of N is generally a randomized algorithm.

[16] Similarly to the MW advantage $(\beta_A - \bar{\beta}_A)^2/(\beta_A + \bar{\beta}_A)$ and statistical distance $(\beta_A - \bar{\beta}_A)$, we define this function as a simple combination of β_A and $\bar{\beta}_A$. We included the $(\beta_A - \bar{\beta}_A)$ factor so that the advantage measure retains the appealing feature that "trivial adversaries" (with $\beta_A = \bar{\beta}_A$) have advantage 0. Our definition is otherwise rather arbitrary.

Theorem 2. *For any decision game $\eth$, the advantage $\mathsf{adv}^{\mathsf{MW}}_{\eth}(A)$ is a convex function of $A \in \mathcal{A}_\perp$, i.e., for any two adversaries $A, B \in \mathcal{A}_\perp$ and $\theta \in (0,1)$, the convex combination $C = \theta \cdot A + (1-\theta) \cdot B \in \mathcal{A}_\perp$ satisfies*

$$\mathsf{adv}^{\mathsf{MW}}_{\eth}(C) \leq \theta \cdot \mathsf{adv}^{\mathsf{MW}}_{\eth}(A) + (1-\theta) \cdot \mathsf{adv}^{\mathsf{MW}}_{\eth}(B).$$

Proof. Using the definition of C, we see that

$$\begin{aligned}\beta_C = \Pr[C(\eth_b) = b] &= \theta \cdot \Pr[A(\eth_b) = b] + (1-\theta) \cdot \Pr[B(\eth_b) = b] \\ &= \theta \cdot \beta_A + (1-\theta) \cdot \beta_B\end{aligned}$$

and similarly for $\bar{\beta}_C$, α_C and δ_C. Therefore, by Lemma 1,

$$\begin{aligned}\mathsf{adv}^{\mathsf{MW}}_{\eth}(C) = \frac{\delta_C^2}{\alpha_C} &= \frac{(\theta \cdot \delta_A + (1-\theta) \cdot \delta_B)^2}{\theta \cdot \alpha_A + (1-\theta) \cdot \alpha_B} \\ &\leq \theta \cdot \frac{\delta_A^2}{\alpha_A} + (1-\theta) \cdot \frac{\delta_B^2}{\alpha_B} \\ &= \theta \cdot \mathsf{adv}^{\mathsf{MW}}_{\eth}(A) + (1-\theta) \cdot \mathsf{adv}^{\mathsf{MW}}_{\eth}(B).\end{aligned}$$

An immediate consequence of convexity is that optimal aborting adversaries $A \in \mathcal{A}_\perp$ can be easily derandomized by fixing the value of the randomness that achieves the highest advantage.

Corollary 1. *For any decision game $\eth$ and (randomized) adversary $A(\cdot; r)$, there is a value of r such that the deterministic adversary $A_r(\cdot) = A(\cdot; r)$ has advantage at least $\mathsf{adv}^{\mathsf{MW}}_{\eth}(A_r) \geq \mathsf{adv}^{\mathsf{MW}}_{\eth}(A)$.*

Proof. Any randomized adversary $A \in \mathcal{A}_\perp$ can be written as a convex combination $A = \sum_r \Pr[r] \cdot A_r$ of deterministic adversaries $A_r(\cdot) = A(\cdot; r)$ indexed by the randomness r. It follows by Theorem 2 that

$$\mathsf{adv}^{\mathsf{MW}}_{\eth}(A) \leq \sum_r \Pr[r] \cdot \mathsf{adv}^{\mathsf{MW}}_{\eth}(A_r) \leq \max_r \mathsf{adv}^{\mathsf{MW}}_{\eth}(A_r). \tag{13}$$

Choosing the value of r that achieves the maximum gives a deterministic adversary A_r with an advantage which is at least as good as A.

Note that the deterministic adversary of Corollary 1 has the same running time $T(A_r) = T(A)$ as the original randomized adversary because we are just fixing the randomness. So, Corollary 1 says that the optimal bit-security is achieved by a deterministic adversary. However, this is only true in a non-uniform setting, where the optimal randomness r can be hardwired in the code of A. In a uniform setting, when considering probability ensembles over larger and larger sets indexed by a security parameter κ, determinining the optimal value of r can be computationally difficult. In particular, trying all possible values of r and estimating which one is best is not computationally feasible because there are exponentially (in κ) possible values of r, and, in any case, it would result again in a randomized adversary. This is the only result in this paper that makes essential use of the non-uniform model.

3.3 Threshold Adversaries Are Optimal

In this subsection we focus on the simple problem of distinguishing between two probability distributions $\mathcal{X} = (X_0, X_1)$ over a set Ω (as opposed to arbitrary distinguishing games), in the statistical security setting, i.e., when the computational cost of the distinguisher is not taken into account. This problem reduces to determining the highest possible advantage $\Delta^2_{\mathsf{MW}}(\mathcal{X}) = \mathsf{adv}^{\mathsf{MW}}_{\mathcal{X}}(A)$ achieved by a (computationally unbounded) adversary A. All our adversaries can be implemented very efficiently given oracle access to the probabilities $\Pr\{X_b = x\}$. So, the results apply to the computational security setting as well when the probability distributions X_0, X_1 are efficiently computable.

In the case of traditional (non-aborting) adversaries, it is well known that this problem admits a very simple, closed form optimal distinguisher

$$A^{\mathcal{X}}_{\mathsf{SD}}(x) = \mathsf{sign}(\Pr\{X_0 = x\} - \Pr\{X_1 = x\})$$

which, on input a sample x, outputs the bit $b \in \{0, 1\}$ such that the probability $\Pr\{X_b = x\}$ is highest. Note that the distinguisher $A^{\mathcal{X}}_{\mathsf{SD}}$ is efficient only when the probabilities $\Pr\{X_0 = x\}, \Pr\{X_1 = x\}$ are efficiently computable. In this subsection we explore if a similar, closed form optimal distinguisher can also be described for the more general aborting $\mathcal{A}_\perp$ and fuzzy $\mathcal{A}_\approx$ adversaries.

The next lemma shows that even for fuzzy distinguishers, the "sign" of the output should be set to $\mathsf{sign}(A(x)) = \mathsf{sign}(\Pr\{X_0 = x\} - \Pr\{X_1 = x\})$ and the only extra freedom afforded by fuzzy adversaries is the choice of the confidence $|A(x)|$.

Lemma 13. *For any $A \in \mathcal{A}_\approx$ and $\mathcal{X} = (X_0, X_1)$, define the modified adversary*

$$\hat{A}(x) = |A(x)| \cdot \mathsf{sign}(\Pr\{X_0 = x\} - \Pr\{X_1 = x\})$$

that on input x outputs the same confidence $|A(x)|$ as A, and fixes the sign of the output to match $A^{\mathcal{X}}_{\mathsf{SD}}(x)$. Then, this modified adversary satisfies $\mathsf{adv}^{\mathsf{MW}}_{\mathcal{X}}(\hat{A}) \geq \mathsf{adv}^{\mathsf{MW}}_{\mathcal{X}}(A)$.

Proof. It is straightforward to verify that $\left|\hat{A}(x; r)\right| \leq |A(x; r)|$. So, the expected confidence of the modified adversary satisfies $\alpha_{\hat{A}} \leq \alpha_A$. We also have

$$\begin{aligned}
|\delta_A| &= \left|\mathbb{E}_{b;r}[(-1)^b \cdot A(X_b; r)]\right| \\
&= \left|\sum_x \mathbb{E}_r[A(x; r) \cdot \frac{\Pr\{X_0 = x\} - \Pr\{X_1 = x\}}{2}]\right| \\
&\leq \sum_x \mathbb{E}_r[|A(x; r)| \cdot \frac{|\Pr\{X_0 = x\} - \Pr\{X_1 = x\}|}{2}] \\
&= \sum_x \mathbb{E}_r[\hat{A}(x; r) \cdot \frac{\Pr\{X_0 = x\} - \Pr\{X_1 = x\}}{2}] \\
&= \left|\mathbb{E}_{b;r}[(-1)^b \cdot \hat{A}(X_b; r)]\right| = \delta_{\hat{A}}.
\end{aligned}$$

It follows that $\mathsf{adv}^{\mathsf{MW}}_{\mathcal{X}}(A) = \delta_A^2/\alpha_A \leq \delta_{\hat{A}}^2/\alpha_{\hat{A}} = \mathsf{adv}^{\mathsf{MW}}_{\mathcal{X}}(\hat{A})$.

Notice that if $A \in \tilde{\mathcal{A}}_\perp$, then $\hat{A} \in \tilde{\mathcal{A}}_\perp$. So, when applied to aborting adversaries, Lemma 13 shows that the adversary achieving the optimal advantage $\Delta^2_{\mathsf{MW}}(\mathcal{X})$ must agree with $A^{\mathcal{X}}_{\mathsf{SD}}$, except possibly for replacing the output with $\perp$ when confidence is low.

At this point we are left with the problem of determining how a fuzzy adversary should set the output confidence, and, as a special case, when an aborting adversary should output $\perp$. To this end, define the function

$$\ell_{\mathcal{X}}(x) = \log \frac{\Pr\{X_0 = x\}}{\Pr\{X_1 = x\}} = \log \Pr\{X_0 = x\} - \log \Pr\{X_1 = x\} \tag{14}$$

and consider the class of adversaries that output $\perp$ when $|\ell_{\mathcal{X}}(x)|$ is below a given threshold. Note that $\left|\ell_{(X_0,X_1)}(x)\right| = \left|\ell_{(X_1,X_0)}(x)\right|$ is symmetric in the ordering of the two distributions, and $\mathsf{sign}(\ell_{\mathcal{X}}(x)) = A^{\mathcal{X}}_{\mathsf{SD}}(x)$ because log is a monotonically increasing function.

Definition 7. *We say that $A \in \mathcal{A}_{\approx}$ is a threshold distinguisher between two distributions $\mathcal{X} = (X_0, X_1)$ over a set Ω if there is a threshold $\tau \geq 0$ such that*[17]

$$A(x) = \begin{cases} 0 & \text{if } |\ell_{\mathcal{X}}(x)| \leq \tau \\ \mathsf{sign}(\ell_{\mathcal{X}}(x)) & \text{if } |\ell_{\mathcal{X}}(x)| > \tau \end{cases}$$

Theorem 3. *Let $\mathcal{X} = (X_0, X_1)$ be a pair of probability distributions on a set Ω. Then, the optimal advantage $\Delta^2_{\mathsf{MW}}(\mathcal{X})$ is achieved by a threshold distinguisher A. Moreover, the threshold*

$$\tau^* = \log\left(\frac{4}{3 - 2\beta^*_A} - 1\right)$$

is a simple function of the conditional success probability $\beta^*_A = \beta_A/\alpha_A$. In particular, as $\beta^*_A \in [1/2, 1]$, the threshold satisfies $\exp(\tau) \in [1, 3]$.

Proof. Let $A \in \mathcal{A}_{\approx}$ be an optimal fuzzy adversary, and assume without loss of generality that $\alpha_A, \delta_A > 0$, i.e., A is non-trivial. Now, fix any point x^* in the support of X_0, X_1, and define

$$\alpha^* = \frac{\Pr\{X_0 = x^*\} + \Pr\{X_1 = x^*\}}{2} > 0$$
$$\delta^* = \frac{|\Pr\{X_0 = x^*\} - \Pr\{X_1 = x^*\}|}{2}.$$

We will prove that

- $\frac{\delta^*}{\alpha^*} \leq \frac{\delta_A}{2\alpha_A}$ if and only if $|\ell_{\mathcal{X}}(x^*)| \leq \tau^*$ (and similarly for $\geq$),
- if $\frac{\delta^*}{\alpha^*} \leq \frac{\delta_A}{2\alpha_A}$ then $|A(x^*)| = 1$, and
- if $\frac{\delta^*}{\alpha^*} \geq \frac{\delta_A}{2\alpha_A}$ then $|A(x^*)| = 0$.

[17] The choice that $A(x) = 0$ when $|\ell_{\mathcal{X}}(x)| = \tau$ is somehow arbitrary. We will use a threshold τ such that $|\ell_{\mathcal{X}}(x)| = \tau$ with probability 0.

In particular, since $|A(x^*)| = 0$ and $|A(x^*)| = 1$ are mutually exclusive, it must be $|\ell_{\mathcal{X}}(x^*)| \neq \tau^*$.

Note that the values α^*, δ^* and τ^* satisfy

$$\frac{\delta_A}{2\alpha_A} = \beta_A^* - \frac{1}{2} = 1 - \frac{2}{\exp(\tau^*) + 1}$$
$$\frac{\delta^*}{\alpha^*} = \frac{\exp(|\ell(x^*)|) - 1}{\exp(|\ell(x^*)|) + 1} = 1 - \frac{2}{\exp(|\ell(x^*)|) + 1}.$$

Since $\tau \mapsto 1 - 2/(\exp(\tau) + 1)$ is is a monotonically increasing function, this proves that $|\ell_{\mathcal{X}}(x^*)| \leq \tau^*$ if and only if $(\delta^*/\alpha^*) \leq \delta_A/(2\alpha_A)$.

Now assume $(\delta^*/\alpha^*) \leq \delta_A/(2\alpha_A)$ and (for contradiction) $|A(x^*)| < 1$. Consider a modified adversary A^* which is identical to A, except that $|A^*(x^*)| = |A(x^*)| + \epsilon$, for some $\epsilon < 1 - |A(x^*)|$. Using the definition of δ_A and α_A, we get $\delta_{A^*} = \delta_A + \epsilon \cdot \delta^*$, and $\alpha_{A^*} = \alpha_A + \epsilon \cdot \alpha^*$. So, this modification increases the advantage of A by

$$\begin{aligned}
\mathsf{adv}_{\mathcal{X}}^{\mathsf{MW}}(A^*) - \mathsf{adv}_{\mathcal{X}}^{\mathsf{MW}}(A) &= \frac{(\delta_A + \epsilon \cdot \delta^*)^2}{\alpha_A + \epsilon \cdot \alpha^*} - \frac{\delta_A^2}{\alpha_A} \\
&= \frac{\epsilon^2(\delta^*)^2 + 2\epsilon\delta_A\alpha_A^*\left(\frac{\delta_A}{2\alpha_A} - \frac{\delta^*}{\alpha^*}\right)}{\alpha_A + \epsilon\alpha^*} \\
&\geq \frac{\epsilon^2(\delta^*)^2}{\alpha_A + \epsilon\alpha^*} > 0.
\end{aligned}$$

This is a contradiction to the optimality of A.

Similarly, if $(\delta^*/\alpha^*) \geq \delta_A/(2\alpha_A)$ and (for contradiction) $|A(x^*)| > 0$, we may define a modified adversary A^* that reduces the confidence $|A^*(x^*)| = |A(x^*)| - \epsilon$ of A on x^* by some $\epsilon < |A(x^*)|$. This increases the advantage of A by

$$\begin{aligned}
\mathsf{adv}_{\mathcal{X}}^{\mathsf{MW}}(A^*) - \mathsf{adv}_{\mathcal{X}}^{\mathsf{MW}}(A) &= \frac{(\delta_A - \epsilon \cdot \delta^*)^2}{\alpha_A - \epsilon \cdot \alpha^*} - \frac{\delta_A^2}{\alpha_A} \\
&= \frac{\epsilon^2(\delta^*)^2 - 2\epsilon\delta_A\alpha_A^*\left(\frac{\delta_A}{2\alpha_A} - \frac{\delta^*}{\alpha^*}\right)}{\alpha_A - \epsilon\alpha^*} \\
&\geq \frac{\epsilon^2(\delta^*)^2}{\alpha_A - \epsilon\alpha^*} > 0,
\end{aligned}$$

again contradicting the optimality of A.

4 Equivalence of MW and WY Bit Security

In [15], it is claimed that for any decision game $\mathfrak{D}$, the quantities $\mathsf{WY}(\mathfrak{D})$ and $\mathsf{MW}(\mathfrak{D})$ are equal up to an additive constant, i.e., the MW and WY notions of bit-security are equivalent. However, [15] only proves the statement for a variant of the WY security definition that uses aborting adversaries (i.e., the MW adversaries with output in $\{0, 1, \perp\}$ introduced in [10]), rather than the traditional

(non-aborting, inner) adversaries used in [14] to define WY security. To close this gap, [15] informally states that changing the class of adversaries does not affect the definition of $\mathsf{WY}(\mathcal{D})$, and justifies the assertion saying that the definition does not *explicitly* depend on the size of the co-domain[18] of the adversary A. However, this reasoning is incorrect because the quantity $\Delta_{1/2}(A(\mathcal{D}))$ used in the definition *implicitly* depends on the size of the co-domain of A. Still, the equivalence claimed in [15] holds true, as shown in the following theorem which gives a direct proof that $\mathsf{WY}_{\mathcal{D}}$ and $\mathsf{MW}_{\mathcal{D}}$ are equivalent.

The theorem makes use of the following technical lemma to modify an aborting adversary in such a way that it uses only two of the output symbols in $\{0, 1, \perp\}$.

Lemma 14. *For any decision game $\mathcal{D} = (\mathcal{D}_0, \mathcal{D}_1)$, and aborting adversary $A \in \mathcal{A}_\perp$, there exists a modified adversary $A' \in \mathcal{A}_\perp$ with output in $\{\hat{b}, \perp\}$ (for some fixed $\hat{b} \in \{0,1\}$) and similar running time $T(A) = T(A')$, such that*

$$\mathsf{adv}^{\mathsf{MW}}_{\mathcal{D}}(A') \geq \tfrac{1}{2} \cdot \mathsf{adv}^{\mathsf{MW}}_{\mathcal{D}}(A).$$

Proof. Let $A' = A^z$ be the modified adversary from Lemma 6 with z the value in $\{0,1\}$ that maximizes the advantage $\mathsf{adv}^{\mathsf{MW}}_{\mathcal{D}}(A^z)$. For $i \in \{0,1\}, j \in \{0,1,\perp\}$, let $p_{i,j} = \Pr\{A(\mathcal{D}_i) = j\}$, so that $\beta_A = (p_{0,0} + p_{1,1})/2$, $\bar{\beta}_A = (p_{0,1} + p_{1,0})/2$ and, by Lemma 6,

$$\mathsf{adv}^{\mathsf{MW}}_{\mathcal{D}}(A^j) = \frac{(p_{0,j} - p_{1,j})^2}{2(p_{0,j} + p_{1,j})}.$$

We can then bound

$$\begin{aligned}
\mathsf{adv}^{\mathsf{MW}}_{\mathcal{D}}(A) &= \frac{(\beta_A - \bar{\beta}_A)^2}{\beta_A + \bar{\beta}_A} \\
&= \frac{1}{2} \frac{((p_{0,0} - p_{1,0}) - (p_{0,1} - p_{1,1}))^2}{(p_{0,0} + p_{1,0}) + (p_{0,1} + p_{1,1})} \\
&\leq \frac{(p_{0,0} - p_{1,0})^2}{2(p_{0,0} + p_{1,0})} + \frac{(p_{0,1} - p_{1,1})^2}{2(p_{0,1} + p_{1,1})} \\
&= \mathsf{adv}^{\mathsf{MW}}_{\mathcal{D}}(A^0) + \mathsf{adv}^{\mathsf{MW}}_{\mathcal{D}}(A^1) \\
&\leq 2\mathsf{adv}^{\mathsf{MW}}_{\mathcal{D}}(A^z)
\end{aligned}$$

where the first inequality is Lemma 1, and the second one follows by our choice of z.

We also need a variant of Lemma 14 which gives a tight connection between the MW advantage and the (squared) Le Cam distance of the adversary output probability distributions $A(\mathcal{D})$. A similar statement was previously proved in [15] under the condition that $\Delta_{1/2}(A(\mathcal{D})) \leq 1$, and with worse multiplicative constants.

[18] Recall that the co-domain of A is the set of all possible outputs of A, e.g., $\{0,1\}$ or $\{0,1,\perp\}$.

Lemma 15. *For any decision game* $\eth = (\eth_0, \eth_1)$ *and aborting adversary* $A \in \mathcal{A}_\perp$*, there is a modified adversary* $A' \in \mathcal{A}_\perp$ *with similar running time* $T(A) \approx T(A')$*, such that*

$$\mathsf{adv}^{\mathsf{MW}}_{\eth}(A) \leq \Delta^2_{\mathsf{LC}}(A(\eth)) \leq 3\mathsf{adv}^{\mathsf{MW}}_{\eth}(A').$$

Proof. The proof proceeds as in Lemma 14, using the same notation, except that this time $\mathsf{adv}^{\mathsf{MW}}_{\mathcal{X}}(A^z)$ is maximized over $z \in \{0, 1, \perp\}$. As in the proof of Lemma 14, we still have

$$\mathsf{adv}^{\mathsf{MW}}_{\mathcal{X}}(A) \leq \mathsf{adv}^{\mathsf{MW}}_{\mathcal{X}}(A^0) + \mathsf{adv}^{\mathsf{MW}}_{\mathcal{X}}(A^1).$$

To prove the new lemma we notice that

$$\Delta^2_{\mathsf{LC}}(A(X_0), A(X_1)) = \sum_{j \in \{0,1,\perp\}} \frac{(p_{0,j} - p_{1,j})^2}{2(p_{0,j} + p_{1,j})} = \sum_{j \in \{0,1,\perp\}} \mathsf{adv}^{\mathsf{MW}}_{\mathcal{X}}(A^j)$$

which is at least $\mathsf{adv}^{\mathsf{MW}}_{\mathcal{X}}(A^0) + \mathsf{adv}^{\mathsf{MW}}_{\mathcal{X}}(A^1)$ and at most $3\mathsf{adv}^{\mathsf{MW}}_{\mathcal{X}}(A^z)$.

Theorem 4. *For any decision game* $\eth$, $\mathsf{WY}(\eth) = \mathsf{MW}(\eth) + \Theta(1)$.

Proof. The inequality $\mathsf{MW}_{\eth} \leq \mathsf{WY}_{\eth}$ was already proved in [14]. Here we prove $\mathsf{WY}_{\eth} \leq \mathsf{MW}_{\eth} + O(1)$. Note that by Lemma 15, Lemma 2 and Lemma 3, for any adversary A, we have

$$\mathsf{adv}^{\mathsf{MW}}_{\eth}(A) \leq \Delta^2_{\mathsf{LC}}(A(\eth)) \leq 2 \cdot \Delta^2_{\mathsf{H}}(A(\eth)) \leq \Delta_{1/2}(A(\eth)).$$

So, by Lemma 14, for any adversary A there is an adversary A' such that

$$\begin{aligned} \mathsf{MW}_{\eth}(A) &= \log_2 \frac{T(A)}{\mathsf{adv}^{\mathsf{MW}}_{\eth}(A)} \\ &\geq \log_2 \frac{T(A')}{2\mathsf{adv}^{\mathsf{MW}}_{\eth}(A')} \\ &\geq \log_2 \frac{T(A')}{\Delta_{1/2}(A^z(\eth))} - 1. \end{aligned}$$

Note that A' has co-domain $\{b, \perp\}$ (rather than $\{0, 1\}$). But since $\Delta_{1/2}$ does not give any special meaning to the symbols output by the adversary, we can view A' as a valid adversary for $\mathsf{WY}_{\eth}$. So, we get that $\mathsf{MW}_{\eth}(A) \geq \mathsf{WY}_{\eth}(A') - 1$. Since A was arbitrary, this proves the theorem.

The previous theorem shows that one can use $\mathsf{WY}(\eth)$ as an alternative characterization of $\mathsf{MW}(\eth)$. This is potentially interesting, as $\mathsf{WY}(\eth)$ only makes use of traditional (non-aborting) adversaries, which are perhaps more intuitive and easier to use. (This was indeed one of the motivations of [14].) In particular, it is tempting to assume that, since the inner adversary of [14] always outputs either 0 or 1 (i.e., it never aborts), the optimal WY advantage in distinguishing

between two distributions $\mathcal{X} = (X_0, X_1)$ is achieved by the maximum likelihood distinguisher $A^{\mathcal{X}}_{\mathsf{SD}}$. Perhaps counterintuitively, the following theorem shows that this is not the case, and even if [14] does not make use of aborts, the obvious (inner) distinguishing strategy $A^{\mathcal{X}}_{\mathsf{SD}}$ is not optimal, and can in fact substantially overestimate the number of bits of security by a factor[19] close to 2.

Theorem 5. *There exist (efficiently samplable, efficiently computable) distributions $\mathcal{X} = (X_0, X_1)$ such that*

$$\mathsf{WY}_{\mathcal{X}}(A^{\mathcal{X}}_{\mathsf{SD}}) \geq 2 \cdot \mathsf{MW}(\mathcal{X}) - O(1).$$

Proof. The choice of $\mathcal{X}$ below is from [13, Lemma 2], where it was used to show the suboptimality of distinguishing a product distribution $\mathcal{X}^{\otimes n} = (X_0^{\otimes n}, X_1^{\otimes n})$ by first computing $A^{\mathcal{X}}_{\mathsf{SD}}$ "coordinate-wise" (sometimes called *Scheffé's test*). Consider the distributions $\mathcal{X} = (X_0, X_1)$ shown in following table, where $\epsilon \leq 1/8$:

	0	1	2
X_0	0.5	$0.5 - \epsilon$	ϵ
X_1	$0.5 - \epsilon$	$0.5 + \epsilon$	0
$A^{\mathcal{X}}_{\mathsf{SD}}$	0	1	0
$A^{\mathcal{X}}_{\mathsf{MW}}$	$\perp$	$\perp$	0
$A^{\mathcal{X}}_{\mathsf{SD}}(X_0)$	$0.5 + \epsilon$	$0.5 - \epsilon$	
$A^{\mathcal{X}}_{\mathsf{SD}}(X_1)$	$0.5 - \epsilon$	$0.5 + \epsilon$	

The table also shows the optimal $A^{\mathcal{X}}_{\mathsf{SD}}$ distinguisher, its output distribution on input X_0 and X_1, and a candidate[20] MW distinguisher which we will use in our proof. The intuition is clear: if the sample is 2, then it certainly comes from distribution X_0, but for the other samples the distinguisher does not have enough confidence to make the call. This distinguisher succeeds with probability $\beta = \epsilon/2$, but it never fails. So, it achieves advantage $(\beta - \bar{\beta})^2/(\beta + \bar{\beta}) = \beta = \epsilon/2$. Since A_{MW} runs in constant time, the decisional problem $\mathcal{X}$ has at most $\log_2(2/\epsilon) = 1 + \log_2(1/\epsilon)$ bits of security.

Let's now estimate the advantage achieved by A_{SD} as an inner distinguisher. We first evaluate the Hellinger distance

$$\Delta^2_{\mathsf{H}}(A^{\mathcal{X}}_{\mathsf{SD}}(X_0), A^{\mathcal{X}}_{\mathsf{SD}}(X_1)) = 1 - \sqrt{1 - 4\epsilon^2} \leq 4\epsilon^2$$

where we have used the inequality $1 - \sqrt{1-x} \leq x$, which is valid for all $x \in [0, 1]$. Finally, using Lemma 3, we bound

$$\Delta_{1/2}(A^{\mathcal{X}}_{\mathsf{SD}}(\mathcal{X})) \leq 4\Delta^2_{\mathsf{H}}(A^{\mathcal{X}}_{\mathsf{SD}}(\mathcal{X})) \leq 16\epsilon^2.$$

Since A_{SD} also runs in constant time, the upper bound on bit security it gives is $\log_2(1/(16\epsilon^2)) = 2\log_2(1/\epsilon) - 4$. In summary, if $\epsilon = 2^{-k}$ (for any $k \geq 3$), the bit security is at most $k+1$, but the WY framework with non-aborting distinguisher A_{SD} only provides a very weak bound of $2k - 4$

[19] This is a doubling of the number of security bits k, so it corresponds to overestimating the cost of the attack by an exponential factor 2^k.

[20] This is indeed the optimal MW distinguisher when $\epsilon \leq 1/8$. When $\epsilon \geq 1/8$, then $A^{\mathcal{X}}_{\mathsf{SD}}$ becomes optimal.

5 A Toolbox for Analysis of (c, s)-Bit Security

In this section we use the close relation between the MW and the Le Cam distance (Lemma 15) to establish two fundamental tools for the use of computational-statistical bit security in the analysis of complex cryptograhic protocols: the hybrid proof technique, and the probability replacement theorem.

Theorem 6. *Let $X_0, \ldots, X_k$ be a sequence of cryptographic games. If for all $i = 1, \ldots, k$, $\mathcal{X}_i = (X_{i-1}, X_i)$ is (c_i, s_i)-bit secure, then $\mathcal{X} = (X_0, X_k)$ is (c, s)-bit secure for*

$$c = \min_i(c_i) - 2\log_2(\sqrt{3}k)$$
$$s = \min_i(s_i) - 2\log_2(\sqrt{3}k)$$

Proof. Using Lemma 15 we get the upper bound

$$\begin{aligned}\sqrt{\mathsf{adv}^{\mathsf{MW}}_{\mathcal{X}}(A)} &\leq \Delta_{\mathsf{LC}}(A(X_0), A(X_k)) \\ &\leq \sum_i \Delta_{\mathsf{LC}}(A(X_i), A(X_{i+1})) \\ &\leq \sqrt{3}\sum_i \max_{z_i} \sqrt{\mathsf{adv}^{\mathsf{MW}}_{\mathcal{X}_i}(A^{z_i})} \\ &\leq \sqrt{3}k\sqrt{\max_i(T(A^{z_i})2^{-c_i}, 2^{-s_i})}.\end{aligned}$$

So, since $T(A) \approx T(A^{z_i})$ for all i, the advantage $\mathsf{adv}^{\mathsf{MW}}_{\mathcal{X}}(A)$ is at most

$$3k^2 \max(T(A)2^{-\min_i c_i}, 2^{-\min_i s_i}) = \max(T(A)2^{-c}, 2^{-s}).$$

This proves that $\mathcal{X}$ is at least (c, s)-secure.

This may be seen as an extension of [10, Theorem 7], which is an analogous result for (c, c)-bit security, though with slightly smaller[21] loss of $\log_2(2k^2) = 2\log_2(\sqrt{2}k)$ bits.

We next establish a distribution replacement theorem for (c, s)-bit security for games $\ominus^Y$ parameterized by a distribution Y. This was done in [10] under the assumption that (Y_0, Y_1) is statistically $((\infty, s)$-bit) secure, and in [15] under the assumption that (Y_0, Y_1) is computationally $((c, c)$-bit) secure. We extend this to a (c, s)-bit security assumption below.

[21] One can recover the exact same loss ($\log_2(2k^2) = 2\log_2(\sqrt{2}k)$) by giving a variant of Lemma 15 with constant factor 2 rather than 3. This can be done by comparing $\mathsf{adv}^{\mathsf{MW}}_{\mathcal{X}}(A)$ to $\Delta^2_{\mathsf{LC}}(X'_0, X'_1)$, where $X'_b \in [0,1]^2$ is the first two coordinates of $A(X_b) \in [0,1]^3$. This is to say that one can exactly generalize [10, Theorem 7] by working with (X'_0, X'_1) that are positive measures of total mass ≤ 1 rather than *probability* measures of total mass $= 1$. We avoid doing this as the quantitative improvement is small, at the cost of a large amount of conceptual overhead.

Theorem 7. *Let $\eth, \mathcal{Y}$ be decision games. If $\eth^{Y_0}$ is (c,s)-bit secure, and $\mathcal{Y}$ is (c',s')-bit secure, then $\eth^{Y_1}$ is (c'',s'')-bit secure, where $c'' = \min(c-2, c'-3-\log_2(1+T_\eth))$, and $s'' = \min(s-2, s'-3)$. In particular, if $\mathcal{Y}$ and $\eth^{Y_0}$ are (c,s)-bit secure and[22] $T_\eth = O(1)$, then $\eth^{Y_1}$ is almost (c,s)-bit secure, up to a small additive constant term in bit security.*

Proof. Let A be any adversary. By Lemma 15 and the triangle inequality (for Δ_{LC}), we have We compute

$$\begin{aligned}\sqrt{\mathsf{adv}^{\mathsf{MW}}_{\eth^{Y_1}}(A)} &\le \Delta_{\mathsf{LC}}(A(\eth_0^{Y_1}), A(\eth_1^{Y_1})) \\ &\le \Delta_{\mathsf{LC}}(A(\eth_0^{Y_1}), A(\eth_0^{Y_0})) + \Delta_{\mathsf{LC}}(A(\eth_0^{Y_0}), A(\eth_1^{Y_0})) + \Delta_{\mathsf{LC}}(A(\eth_1^{Y_0}), A(\eth_1^{Y_1})).\end{aligned}$$

We bound each term in the last sum separately. For the middle term, using the upper bound in Lemma 15 and $T(A) = T(A^z)$, we get

$$\Delta_{\mathsf{LC}}(A(\eth_0^{Y_1}), A(\eth_0^{Y_0})) \le \sqrt{3 \max_z \mathsf{adv}^{\mathsf{MW}}_{\eth^{Y_0}}(A^z)} \le \sqrt{3\max(T_A 2^{-c}, 2^{-s})}$$

The other terms are bound constructing distinguishers A_0, A_1 against the game $\mathcal{Y}$ as follows. A_0^Y simulates the execution of A in the game $\eth_0^Y$ and flips the answer, i.e., it outputs $1-a$ when A outputs $a \in \{0,1\}$, and $\perp$ otherwise. A_1^Y simulates the execution of A in the game $\eth_1^Y$, and outputs the same result as A. Then, we have

$$\begin{aligned}\Delta_{\mathsf{LC}}(A(\eth_0^{Y_1}), A(\eth_0^{Y_0})) &= \Delta_{\mathsf{LC}}(A_0(Y_0), A_0(Y_1)) \\ &\le \sqrt{3 \max_z \mathsf{adv}^{\mathsf{MW}}_{\mathcal{Y}}(A_0^z)} \\ &\le \sqrt{3\max(T(A)(1+T_\eth)2^{-c'}, 2^{-s'})}\end{aligned}$$

and similarly for the last term $\Delta_{\mathsf{LC}}(A(\eth_1^{Y_0}), A(\eth_1^{Y_0}))$ using adversary A_1. Combining the three terms gives the bound in the theorem.

6 Conclusion and Open Problems

We developed a number of useful tools to evaluate the bit security of decisional cryptographic properties, in the statistical and computational setting, or even combinations of the two. These include a characterization of the structure of the optimal statistical "aborting" adversaries to facilitate the use of approximate probability distributions (like uniform or discrete gaussians), and general hybrid arguments and probability replacement theorems to combine subprotocols together and support modular security analysis. More tools may be added to the toolbox in the future, but we believe that the results presented in this

[22] Recall from Definition 1 that $T_\eth$ is the relative running time of $\eth$. So, $T_\eth = O(1)$ is quite common, e.g., when oracle calls can be answered in linear time.

paper already demonstrate that computational-statistical bit-security can be quite usable and useful.

For all results in this paper we focused on decision problems, which are the hardest case, but combining decisional primitives with search ones should be fairly straightforward, as the definition of bit security for search problems is standard. An interesting direction for future work is to explore the space between search and decision problems. These include, for example, problems with small (polynomially sized) search space, like password authenticated key exchange. Two works [6,10] offer general definitions that interpolate between search and decision problems, but the significance of those definitions for intermediate problems is unclear. Similarly to what was done in [10] for search and decision problems, it would be interesting to analyze a representative set of protocols falling in-between search and decision primitives, possibly in conjunction with standard search and decision primitives, to see if the bit-security estimates provided by those definitions match the cryptographic intuition behind the informal notion of bit-security.

Another interesting direction for further work is to make good use of the definition of computational-statistical bit-security (proposed in [8] and studied in this work) to formally analyze concrete protocols of practical interest, and make provable (still tight) claims about their security.

References

1. Abla, P., Liu, F., Wang, H., Wang, Z.: Ring-based identity based encryption - asymptotically shorter MPK and tighter security. In: Nissim, K., Waters, B. (eds.) Theory of Cryptography - 19th International Conference, TCC 2021, Raleigh, NC, USA, November 8-11, 2021, Proceedings, Part III. Lecture Notes in Computer Science, vol. 13044, pp. 157–187. Springer (2021). https://doi.org/10.1007/978-3-030-90456-2_6
2. Bernstein, D.J., Lange, T.: Non-uniform cracks in the concrete: The power of free precomputation. In: Sako, K., Sarkar, P. (eds.) Advances in Cryptology - ASIACRYPT 2013 - 19th International Conference on the Theory and Application of Cryptology and Information Security, Bengaluru, India, December 1-5, 2013, Proceedings, Part II. Lecture Notes in Computer Science, vol. 8270, pp. 321–340. Springer (2013). https://doi.org/10.1007/978-3-642-42045-0_17
3. De, A., Trevisan, L., Tulsiani, M.: Time space tradeoffs for attacks against one-way functions and PRGs. In: Rabin, T. (ed.) Advances in Cryptology - CRYPTO 2010, 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings. Lecture Notes in Computer Science, vol. 6223, pp. 649–665. Springer (2010). https://doi.org/10.1007/978-3-642-14623-7_35
4. Grigoryan, N., Harutyunyan, A., Voloshynovskiy, S., Koval, O.: On multiple hypothesis testing with rejection option. In: 2011 IEEE Information Theory Workshop, pp. 75–79. IEEE (2011)
5. Lalitha, A., Javidi, T.: On error exponents of almost-fixed-length channel codes and hypothesis tests. arXiv preprint arXiv:2012.00077 (2020)
6. Lee, K.: Bit security as cost to observe advantage: Towards the definition from the book. Communications in Cryptology (2024). https://doi.org/10.62056/an5txol7

7. Levin, L.A.: Randomness and non-determinism. J. Symb. Log. **58**, 1102–1103 (1993)
8. Li, B., Micciancio, D., Schultz, M., Sorrell, J.: Securing approximate homomorphic encryption using differential privacy. In: Dodis, Y., Shrimpton, T. (eds.) Advances in Cryptology – CRYPTO 2022, Part I. Lecture Notes in Computer Science, vol. 13507, pp. 560–589. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 15–18, 2022). https://doi.org/10.1007/978-3-031-15802-5_20
9. Micciancio, D., Walter, M.: Gaussian sampling over the integers: efficient, generic, constant-time. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017. LNCS, vol. 10402, pp. 455–485. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63715-0_16
10. Micciancio, D., Walter, M.: On the bit security of cryptographic primitives. In: Nielsen, J.B., Rijmen, V. (eds.) Advances in Cryptology – EUROCRYPT 2018, Part I. Lecture Notes in Computer Science, vol. 10820, pp. 3–28. Springer, Heidelberg, Germany, Tel Aviv, Israel (Apr 29–May 3, 2018). https://doi.org/10.1007/978-3-319-78381-9_1
11. Pensia, A., Jog, V., Loh, P.L.: Communication-constrained hypothesis testing: Optimality, robustness, and reverse data processing inequalities. IEEE Transactions on Information Theory (2023)
12. Polyanskiy, Y., Wu, Y.: Information theory: From coding to learning. Book draft (2022)
13. Suresh, A.T.: Robust hypothesis testing and distribution estimation in Hellinger distance. In: International Conference on Artificial Intelligence and Statistics, pp. 2962–2970. PMLR (2021)
14. Watanabe, S., Yasunaga, K.: Bit security as computational cost for winning games with high probability. In: Tibouchi, M., Wang, H. (eds.) Advances in Cryptology – ASIACRYPT 2021, Part III. Lecture Notes in Computer Science, vol. 13092, pp. 161–188. Springer, Heidelberg, Germany, Singapore (Dec 6–10, 2021). https://doi.org/10.1007/978-3-030-92078-4_6
15. Watanabe, S., Yasunaga, K.: Unified view for notions of bit security. In: Guo, J., Steinfeld, R. (eds.) Advances in Cryptology - ASIACRYPT 2023 - 29th International Conference on the Theory and Application of Cryptology and Information Security, Guangzhou, China, December 4-8, 2023, Proceedings, Part VI. Lecture Notes in Computer Science, vol. 14443, pp. 361–389. Springer (2023). https://doi.org/10.1007/978-981-99-8736-8_12
16. Yasunaga, K.: Replacing probability distributions in security games via Hellinger distance. In: Tessaro, S. (ed.) 2nd Conference on Information-Theoretic Cryptography, ITC 2021, July 23-26, 2021, Virtual Conference. LIPIcs, vol. 199, pp. 17:1–17:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2021). https://doi.org/10.4230/LIPICS.ITC.2021.17

Low-Degree Security of the Planted Random Subgraph Problem

Andrej Bogdanov[1(✉)], Chris Jones[2], Alon Rosen[2], and Ilias Zadik[3]

[1] University of Ottawa, Ottawa, Canada
abogdano@uottawa.ca
[2] Bocconi University, Milan, Italy
{chris.jones,alon.rosen}@unibocconi.it
[3] Yale University, New Haven, USA
ilias.zadik@yale.edu

Abstract. The planted random subgraph detection conjecture of Abram et al. (TCC 2023) asserts the pseudorandomness of a pair of graphs (H, G), where G is an Erdős-Rényi random graph on n vertices, and H is a random induced subgraph of G on k vertices. Assuming the hardness of distinguishing these two distributions (with two leaked vertices), Abram et al. construct communication-efficient, computationally secure (1) 2-party private simultaneous messages (PSM) and (2) secret sharing for forbidden graph structures.

We prove the low-degree hardness of detecting planted random subgraphs all the way up to $k \leq n^{1-\Omega(1)}$. This improves over Abram et al.'s analysis for $k \leq n^{1/2-\Omega(1)}$. The hardness extends to r-uniform hypergraphs for constant r.

Our analysis is tight in the distinguisher's degree, its advantage, and in the number of leaked vertices. Extending the constructions of Abram et al., we apply the conjecture towards (1) communication-optimal multiparty PSM protocols for random functions and (2) bit secret sharing with share size $(1+\epsilon)\log n$ for any $\epsilon > 0$ in which arbitrary minimal coalitions of up to r parties can reconstruct and secrecy holds against all unqualified subsets of up to $\ell = o(\epsilon \log n)^{1/(r-1)}$ parties.

1 Introduction

In the planted clique model [Jer92, Kuc95] one observes the union of an Erdős-Rényi random graph $G_0 \sim G(n, 1/2)$ and a randomly placed $k = k_n$-clique H, i.e., the graph $G = G_0 \cup H$. The goal of the planted clique detection task is to distinguish between observing G from the planted clique model and G which is simply an instance of $G(n, 1/2)$. The planted clique conjecture states that the planted clique instance remains pseudorandom whenever $k \leq n^{1/2-\Omega(1)}$ up to $n^{-\Omega(1)}$ distinguishing advantage. Conversely, multiple polynomial-time algorithms can distinguish with high probability whenever $k = \Omega(\sqrt{n})$. Research on the planted clique conjecture has gone hand-in-hand with key developments in average-case complexity theory over the last decades, including spectral and tensor algorithms [AKS98, FK08], lower bound techniques for restricted

E. Boyle and M. Mahmoody (Eds.): TCC 2024, LNCS 15365, pp. 255–275, 2025.
https://doi.org/10.1007/978-3-031-78017-2_9

classes including the sum-of-squares hierarchy [BHK+19], low-degree polynomial methods [Hop18], statistical query methods [FGR+17] and MCMC methods [Jer92, GZ19, CMZ23], and the development of new average-case reductions [BB20, HS24].

At this point, the conjectured hardness of the planted clique problem around $k \approx \sqrt{n}$ stands as a central conjecture in average-case complexity. But despite its popularity, the cryptographic applications have been quite limited, with one exception in the symmetric-key setting proposed by Juels and Peinado [JP97]. Recently Abram et al. [ABI+23] revisited the planted clique problem and showed how it can be useful in the context of secret sharing and secure computation. The authors specifically show that (slight variants of) the planted clique conjecture can be used to construct a computationally secure scheme whose share size is much smaller than the best existing information-theoretically secure scheme.

In order to obtain further improvements to the share size, Abram et al. proposed a new intriguing conjecture similar to planted clique. They start by defining the following general model (also introduced in [Hul22]).

Definition 1. (Planted (induced) subgraph model[1]) *Fix H to be an arbitrary unlabeled subgraph on k vertices. Then G is chosen to be a random n-vertex graph where a copy of H is placed on k vertices chosen uniformly at random (as an induced subgraph on the k vertices), and all edges without both endpoints on the k vertices appear with probability 1/2.*

When H is the k-clique, the planted subgraph model becomes exactly the planted clique model. The clique is the most structured graph possible and it is natural to wonder:

could the problem be significantly harder if a different graph H is planted?

Abram et al. suggest studying the planted random subgraph model in which H is an instance of $G(k, 1/2)$. An equivalent definition is the following.

Definition 2. (Planted random subgraph model) *One observes a pair (H, G), where G is a random n-vertex graph and H is a random k-subgraph of G with the vertex labels removed.*

Abram et al. make the following interesting conjecture.

Conjecture 1. (Planted Random Subgraph conjecture [ABI+23]) *The planted random subgraph problem is hard up to advantage $n^{-\Omega(1)}$ provided $k \leq n^{1-\Omega(1)}$, with high probability over $H \sim G(k, 1/2)$ as n grows to infinity.*

This stands in contrast to the case that H is a k-clique where a computational phase transition is expected to take place at the smaller value $k \approx n^{1/2}$.

[1] A similar yet different model where one observes the *union* of a copy of H with an instance of $G(n, 1/2)$ has also been recently analyzed in the statistical inference literature [Hul22, MNWS+23, YZZ24]. For this work, we solely focus on the "induced" variant, where H appears as an induced subgraph of G..

Abram et al. confirm the planted random subgraph conjecture in the low-degree analysis framework (to be described below) but only up to the "planted clique threshold" $k \leq n^{1/2-\Omega(1)}$ (a result also independently proven by Huleihel [Hul22]). Their work leaves open the regime $n^{1/2-\Omega(1)} \leq k \leq n^{1-\Omega(1)}$, and in particular the question of whether there is a larger window of hardness for planted random subgraph than for planted clique.

Our main contribution is the confirmation of Conjecture 1 in the low-degree framework. We prove that the planted random subgraph problem remains hard for low-degree distinguishers of degree at most $o((\log n/\log\log n)^2)$ in the *full range* $k \leq n^{1-\Omega(1)}$. The degree is best possible up to $\log\log n$ factors, and the analysis extends also to the case of hypergraphs. See Sect. 2 for the precise theorem statement.

1.1 Secret Sharing and Leakage

For their intended cryptographic applications Abram et al. rely on a strengthening of the planted random subgraph conjecture which also allows for leaked additional information about the embedding of H in G. It is easiest to motivate these stronger conjectures through their intended application.

A *(partial) access structure* for k parties is a pair of set systems R, S over $\{1, \ldots, k\}$, where R is upward-closed, S is downward-closed, and R, S are disjoint. A bit secret sharing scheme consists of a randomized sharing algorithm that maps the secret bit $s \in \{0, 1\}$ into k shares so that sets in R can reconstruct s from their shares with probability one, while sets in S cannot distinguish $s = 0$ or $s = 1$.

In a forbidden graph access structure, R is the edge-set of a graph and S is the union of its complement $\{\{u, v\} \notin R \colon u \neq v \in [k]\}$ and the set $[k]$ of vertices. Abram et al. propose the following secret sharing scheme for any such structure:

Construction 1. Forbidden graph secret sharing:

1. The dealer samples a random n-vertex graph G and remembers a secret k-vertex subgraph H of it randomly embedded via $\phi \colon V(H) \to V(G)$.
2. The dealer publishes the pair (H_s, G), where H_s is a k-vertex graph with adjacency matrix

$$H_s(u, v) = \begin{cases} H(u, v) \oplus s, & \text{if } \{u, v\} \in R \\ \text{a random bit}, & \text{otherwise.} \end{cases} \tag{1}$$

3. The share of party v is the value $\phi(v) \in [n]$.

If $\{u, v\} \in R$, the parties reconstruct by calculating

$$H_s(u, v) \oplus G(\phi(u), \phi(v)) = H(u, v) \oplus G(\phi(u), \phi(v)) \oplus s = s. \tag{2}$$

Secrecy requires that the joint distribution $(H_s, G, \phi(u), \phi(v))$ of the public information and the shares is indistinguishable between $s = 0$ and $s = 1$ provided

$\{u, v\} \in S$. In the absence of the "leakage" $(\phi(u), \phi(v))$ this is a consequence of the planted random subgraph conjecture (Conjecture 1).

To handle the leakage, we consider the following generalization. Two parties $\{u, v\} \in S$ know the location of their edge $H(u, v) = G(\phi(u), \phi(v))$ in G, which could potentially be useful to search for the "local structure of H" around their edge. The new conjecture posits that if u and v have this additional information, they still cannot distinguish whether H is planted. With an eye towards stronger security we state it below for a general ℓ.

Conjecture 2. (Planted random subgraph conjecture with ℓ-vertex leakage) *With high probability over $H \sim G(k, 1/2)$, the following two distributions are $n^{-\Omega(1)}$-indistinguishable in polynomial time for all subsets $L = \{u_1, \ldots, u_\ell\} \subseteq V(H)$ of size ℓ:*

1. *(planted) $(H, G, \phi(u_1), \ldots, \phi(u_\ell))$ where we choose uniformly at random an injective function $\phi : [k] \to [n]$ and embed H into G on the image of ϕ. The remaining edges of G are sampled randomly.*
2. *(model) $(H, G, \phi(u_1), \ldots, \phi(u_\ell))$ where we choose uniformly at random an injective function $\phi : L \to [n]$ and embed the subgraph of H on L into G on the image of ϕ. The remaining edges of G are sampled randomly.*

Assuming this conjecture with $\ell = 2$, given $(\phi(u), \phi(v))$ for $\{u, v\} \in S$, we claim that both (H_0, G) and (H_1, G) are pseudorandom and hence indistinguishable: As $\{u, v\} \in S$, the (u, v)-th bits of H_0 and H_1 in (H_s, G) are independent of all the others and cannot be used to distinguish. Once the (u, v)-th bits of H_0 and H_1 are removed, both (H_0, G) and (H_1, G) become identically distributed to the planted (H, G) with its (u, v)-th bit removed. By the conjecture, this model is indistinguishable from a uniformly random string.

The share size in this scheme is $(1 + o(1)) \log k$. In contrast, the most compact known forbidden graph scheme with perfect security has shares of size $\exp \tilde{\Theta}(\sqrt{\log k})$ [LVW17, ABF+19]. Statistical security requires shares of size $\log k - O(1)$ when R is the complete graph [ABI+23]. It is not known if computational security is subject to the same limitation.

Under the ℓ-vertex leakage assumption the secrecy holds not only against pairs of parties that are not an edge in R, but also against all independent sets up to size ℓ, i.e.,

$$S = \{I \colon I \text{ is an independent set of } R \text{ and } |I| \leq \ell\}.$$

By passing to r-hypergraphs instead of graphs, we naturally extend the construction to R which is an arbitrary subset of at most r parties, with security against all size-ℓ independent sets of R (see Construction 3 below). The most compact known perfectly secure forbidden r-hypergraph scheme has share size $\exp \tilde{\Theta}(\sqrt{r \log k})$ [LVW17] whereas our share size is still $(1 + o(1)) \log k$.

It would be interesting to obtain a provable separation in share size between the computationally secure Construction 3 and the best possible perfectly secure construction for some access structure. In Sect. 4.1 we explain why this is challenging using available methods.

1.2 Private Simultaneous Messages (PSM)

In a PSM, Alice and Bob are given inputs x, y to a public function $F\colon [k]^2 \to \{0,1\}$. They calculate messages $\phi(x), \phi(y)$ which are securely forwarded to Carol. Carol needs to output the value $F(x,y)$ without learning any information about x and y beyond this value.

Abram et al. propose the following PSM protocol. In a setup phase, F viewed as a bipartite graph is randomly embedded into an otherwise random host graph G via ϕ. The graph G is given to Carol and the embedding ϕ is given to Alice and Bob. Carol outputs $G(\phi(x), \phi(y))$ which must equal $F(x,y)$.

Abram et al. argue that this protocol is "secure" for a $(1 - o(1))$-fraction of functions F under Conjecture 2 with leakage $\ell = 2$. Their security definition appears to additionally assume that the choice of inputs (x, y) is independent of the function F. In contrast, our security definition in Sect. 4.2 allows for Alice and Bob to choose their inputs jointly from some distribution that depends on the description of F. This is more natural for potential cryptographic applications; Alice and Bob should not be expected to commit to their input before they know which function they are computing. We extend our low-degree analysis to support this stronger notion of security.

Messages in this protocol are of length $\log n = (1+\epsilon)\log k$. In contrast, perfect security is known to require combined message length $|\phi(x)| + |\phi(y)| \geq (3 - o(1))\log k$ [FKN94, AHMS18] (but it is not known if statistical security is subject to the same bound).

The r-hypergraph variant of the conjecture with leakage $\ell = r$ gives PSM security for r-party protocols also with message size $\log n = (1+\epsilon)\log k$ (Sect. 4.2). Even without a security requirement the message size must be at least $(1 - o(1))\log k$ for the protocol to be correct on most inputs.

1.3 Low-Degree Lower Bounds

We provide evidence for these conjectures in the form of lower bounds against the low-degree polynomial computational model (see e.g., [KWB19] and references therein). In this model, fixing a parameter $D = D_n$, the distinguishing algorithm is allowed to compute an arbitrary degree-D polynomial function of the bits of the input over the field $\mathbb{R}$. The algorithm succeeds if the value of the polynomial is noticeably different between the random and planted models. Degree-D polynomials serve as a proxy for $n^{O(D)}$ time computation since a degree-D polynomial in $\mathrm{poly}(n)$ input bits can be evaluated by brute force in time $n^{O(D)}$ (ignoring numerical issues).

Surprisingly, for noise-robust[2] hypothesis testing problems it has been conjectured that whenever all degree-D polynomials with $D = O(\log n)$ fail (formally, no polynomial strongly separates the two distributions [COGHK+22, Sect. 7]), then no polynomial-time distinguisher succeeds. This is now known as the "low-degree conjecture" of Hopkins [Hop18]. Based on this heuristic, a provable failure

[2] Noise-robustness means that the planted structure is resilient to small random perturbations [Hop18, HW21].

of $O(\log n)$-degree polynomials to strongly separate the two distributions provides a state-of-the-art prediction of the hard and easy regimes for the problem of interest.

It should be noted that there exists a certain weakness in existing low-degree hardness evidence for the planted clique problem, which also applies to our lower bound for the planted random subgraph problem (and that of [ABI+23]). Both planted clique and planted random subgraph technically do not satisfy the noise-robust assumption of the low-degree conjecture because the planted isomorphic copy of H in the graph G is not robust to small perturbations of G (if 0.01 fraction of the edges of G are randomly flipped then the copy of H will be destroyed). Noise-robustness is an important assumption; in fact, in a handful of carefully chosen noise-free problems, low-degree methods are provably weaker than other brittle polynomial-time methods such as Gaussian elimination or lattice-basis reduction techniques [ZSWB22]. That being said, the existing techniques do not appear applicable to graph settings such as planted clique or the planted random subgraph model.

2 Our Result

Let H be an r-uniform hypergraph over vertex set $[k]$ chosen uniformly at random (i.e., each r-hyperedge between the vertices of $[k]$ is included independently with probability half). Let $L \subseteq V(H)$ of size ℓ. Let $\mathcal{P}_{H,L}$ and $\mathcal{Q}_{H,L}$ be the following distributions over r-uniform hypergraphs G with vertex set $[n]$, where $n \geq k \geq \ell$:

1. In the *planted distribution* $\mathcal{P}_{H,L}$, an injective map $\phi\colon [k] \to [n]$ is chosen uniformly at random among all injective maps conditioned on $\phi(u) = u$ for $u \in L$. The hyperedges of G are

$$G(u_1, \ldots, u_r) = \begin{cases} H(\phi^{-1}(u_1), \ldots, \phi^{-1}(u_r)), & \text{if } \phi^{-1}(u_1), \ldots, \phi^{-1}(u_r) \text{ exist} \\ \text{a random bit}, & \text{otherwise.} \end{cases}$$

2. In the *null distribution* $\mathcal{Q}_{H,L}$, the hyperedges of G are

$$G(u_1, \ldots, u_r) = \begin{cases} H(u_1, \ldots, u_r), & \text{if } u_1, \ldots, u_r \in L \\ \text{a random bit}, & \text{otherwise.} \end{cases}$$

Uniform r-hypergraphs on n vertices are represented by their adjacency maps $\binom{[n]}{r} \to \{\pm 1\}$, with -1 and 1 representing the presence and absence of a hyperedge, respectively.

In words, the hypergraph $G \sim \mathcal{P}_{H,L}$ drawn from the planted model has the public hypergraph H embedded into a uniform choice of k vertices, and is otherwise purely random. However, the location of $L \subseteq V(H)$ is fixed and public information. The hypergraph $G \sim \mathcal{Q}_{H,L}$ drawn from the random model copies the subgraph of H on L, but it does not use the part of H outside of L; all remaining edges of the graph are chosen purely at random. Note that in both

models, the marginal distribution of G is a uniformly random hypergraph, but distinguishers know H and L.

In the case $r = 2$ of graphs, there is a slight difference between the distributions $\mathcal{P}_{H,L}, \mathcal{Q}_{H,L}$ and those described in the Introduction, namely that we have imposed the condition $\phi(u) = u$ on the leaked vertices in L. This condition is without loss of generality, and in particular, it does not affect the complexity of distinguishing $\mathcal{P}_{H,L}$ from $\mathcal{Q}_{H,L}$.

Following the low-degree framework [KWB19], we consider the degree-D-likelihood ratio $\mathcal{LR}_D(H, L)$,

$$\mathcal{LR}_D(H,L) = \sup_{\substack{p \in \mathbb{R}[G(\boldsymbol{u}):\boldsymbol{u} \in \binom{[n]}{r}] \\ \deg p \le D}} \mathrm{Adv}_p(H,L)$$

where

$$\mathrm{Adv}_p(H,L) = \frac{\mathbf{E}_{\mathcal{P}_{H,L}}[p(G)] - \mathbf{E}_{\mathcal{Q}_{H,L}}[p(G)]}{\sqrt{\mathbf{Var}_{\mathcal{Q}_{H,L}}[p(G)]}}.$$

Here $p \in \mathbb{R}[G(\boldsymbol{u}) : \boldsymbol{u} \in \binom{[n]}{r}]$ denotes a multivariate polynomial in the quantities $G(u_1, \ldots, u_r)$ for $(u_1, \ldots, u_r) \in \binom{[n]}{r}$ with degree at most D. $\mathcal{LR}_D(H, L)$ measures the best advantage of a degree-D polynomial distinguisher that can arbitrarily preprocess H and knows L. Whenever $\mathcal{LR}_D(H, L) = o(1)$ then no D-degree polynomial can achieve strong separation between $\mathcal{P}_{H,L}$ and $\mathcal{Q}_{H,L}$ [COGHK+22, Sect. 7].

To gain intuition on the performance of low-degree polynomials, let us start with the simplest one, which is the bias of the edges of the hypergraph G:

$$p(G) = \sum_{1 \le u_1 < \cdots < u_r \le n} G(u_1, \ldots, u_r).$$

Assume for simplicity that $L = \emptyset$. It holds by direct expansion,

$$\begin{aligned} \mathbf{E}_{\mathcal{P}_H}[p(G)] &= \sum_{1 \le u_1 < \cdots < u_r \le k} H(u_1, \ldots, u_r) \\ \mathbf{E}_{\mathcal{Q}_H}[p(G)] &= 0 \\ \mathbf{Var}_{\mathcal{Q}_H}[p(G)] &= \tbinom{n}{r}. \end{aligned}$$

The likelihood ratio is

$$\mathrm{Adv}_p(H) = \Theta\left(\frac{\mathbf{E}_{\mathcal{P}_H}[p(G)]}{n^{r/2}}\right).$$

As $\mathbf{E}_{\mathcal{P}_H}[p(G)]$ is a sum of the $\binom{k}{r}$ hyperedge indicators for H, $\mathbf{E}_{\mathcal{P}_H}[p(G)]$ would have value $\pm\Theta(k^{r/2})$ for a typical choice of H, resulting in an advantage of $\Theta((k/n)^{r/2})$ (after optimizing between $p(G)$ or $-p(G)$). The advantage is $o(1)$ when $k \le n^{1-\Omega(1)}$ and therefore the distinguisher fails in this regime. Yet, when $k = \Theta(n)$ the calculation suggests the count distinguisher succeeds with $\Omega_r(1)$ probability which indeed can be confirmed by being a bit more careful in the above analysis. Our main theorem shows that other low-degree polynomials cannot substantially improve upon the edge-counting distinguisher.

Theorem 1. *Assume for some $p \in \mathbb{N}$ and constant $\epsilon > 0$, the following bounds hold on the size of H, k, the leakage number ℓ and the degree D:*

1. $k \leq (n-\ell)n^{-\epsilon}/24p^2D^2 + \ell$
2. $\ell \leq \min\{k,\ \epsilon^{1/(r-1)} r(\log n)^{1/(r-1)}/40\}$ *and,*
3. $D \leq \epsilon^3 (\log n)^{r/(r-1)} / \left(\frac{r}{r-1} \log\log n\right)$.

Then for any $L \subseteq [k]$ with $|L| = \ell$,

$$\left(\mathbf{E}_H \mathcal{LR}_D(H,L)^{2p}\right)^{1/p} \leq \frac{2^{\binom{\ell}{r-1}} n^{-\epsilon}}{1-n^{-\epsilon/2}} + \exp\left(-\Omega\left(r(\epsilon \log n)^{1+1/(r-1)}\right)\right).$$

In particular, for $p = 1$, $\ell = o((\log n)^{1/(r-1)})$, and $\epsilon = \Omega(1)$

$$\mathbf{E}_H \mathcal{LR}_D(H,L)^2 = n^{-\epsilon+o(1)}.$$

The bound is tight in the following ways:

1. **Degree:** The bound on D is optimal (for constant ϵ) up to a factor of $O(\log\log n)$. A degree-$O((r\log n)^{r/(r-1)})$ distinguisher with high advantage and time complexity $2^{O((r\log n)^{1/(r-1)})}$ exists. This is the algorithm that looks for the presence of a subgraph in G that is identical to the one induced by the first $O(r^{r/(r-1)}(\log n)^{1/(r-1)})$ vertices in H.
2. **Leakage:** When $\binom{\ell}{r-1} \geq \log(2n)$ the distinguishing advantage is constant (for any $k > \ell$). The distinguisher that looks for the existence of a vertex in G whose adjacencies in L match those of an arbitrary vertex in H outside L has constant advantage, degree $\binom{\ell}{r-1}$, and time complexity $O(n\binom{\ell}{r-1})$.
3. **Advantage:** The edge-counting distinguisher described above has advantage $(k/n)^{r/2} = n^{-\epsilon r/2}$. Our proof can show a matching lower bound in the absence of leakage. When leakage is present, assuming $\ell > r-1$, the linear distinguisher

$$\operatorname{sign} \sum_{v \notin L} G(1,\ldots,r-1,v) = \operatorname{sign} \sum_{v \notin L} H(1,\ldots,r-1,v)$$

has squared advantage $\Omega((k-\ell)/(n-\ell)) = \Omega(n^{-\epsilon})$ which matches the theorem statement.

2.1 Our Proof

Abram et al. obtain their result as a consequence of a worst-case bound for arbitrary planted H: They prove that for all graphs H with $k \leq n^{1/2-\epsilon}$ vertices,

$$\mathcal{LR}_D(H,L) \leq o(1).$$

As $k = n^{1/2}$ is tight for clique their method cannot prove a better bound. In contrast, we average the likelihood ratio over the choice of H, showing that $\mathbf{E}_H[\mathcal{LR}_D(H,L)^2]$ is small all the way up to $k \leq n^{1-\epsilon}$. By taking the expectation

over H, we introduce extra cancellations that are necessary to obtain the stronger bound.

By Markov's inequality

$$\mathbf{P}_H[\mathcal{LR}_D(H,L)^2 \geq \eta] \leq \frac{\mathbf{E}_H[\mathcal{LR}_D(H,L)^2]}{\eta}\,.$$

A vanishing expectation implies concentration, namely $\mathcal{LR}_D(H,L) = o(1)$ for a $1-o(1)$ fraction of H.

The above calculation bounds the advantage for a fixed leakage set L. In order to bound the advantage of an arbitrary set L for the cryptographic applications, we also bound the higher moments of $\mathcal{LR}_D(H,L)$. Using $p = \ell \log n$ and applying Markov's inequality with $\eta = 4n^{-\epsilon+o(1)}$

$$\begin{aligned}\mathbf{P}_H[\mathcal{LR}_D(H,L)^2 \geq \eta] &\leq \frac{\mathbf{E}_H[\mathcal{LR}_D(H,L)^{2p}]}{\eta^p} \\ &\leq \left(\frac{n^{-\epsilon+o(1)}}{\eta}\right)^p \\ &= 4^{-\ell\log n} \leq \frac{1}{n\binom{n}{\ell}}\,.\end{aligned}$$

Taking a union bound over the $\binom{k}{\ell}$ choices for L, we can deduce the stronger result that no leakage set L can attain advantage η:

$$\mathbf{P}_H\left[\max_{\substack{L\subseteq V(H)\\|L|=\ell}} \mathcal{LR}_D(H,L)^2 \geq 4n^{-\epsilon+o(1)}\right] \leq o(1)\,.$$

We summarize the final bound on the low-degree advantage for Conjecture 2 as the following corollary, which includes the parameters.

Corollary 1. *For all $p \in \mathbb{N}$ and $\eta > 0$,*

$$\begin{aligned}&\mathbf{P}_H\left[\max_{\substack{L\subseteq V(H)\\|L|=\ell}} \mathcal{LR}_D(H,L)^2 \geq \eta\right] \\ &\qquad\leq \binom{n}{\ell}\eta^{-p}\left(\frac{2^{\binom{\ell}{r-1}}n^{-\epsilon}}{1-n^{-\epsilon/2}} + \exp\left(-\Omega\left(r(\epsilon\log n)^{1+1/(r-1)}\right)\right)\right)^p\,.\end{aligned}$$

3 Proof of Theorem 1

Viewed as an $\binom{n}{r}$-dimensional vector, every G in the support of $\mathcal{Q}_{H,L}$ decomposes as (G', G_L), where G_L is the subgraph of G on L and G' is the remaining part (indexed by r-subsets that have at least one vertex in $[n] \setminus L$).

We start by claiming that without loss of generality, all polynomial distinguishers of interest are constant in the coordinates of G_L. Indeed, in both the planted $\mathcal{P}_{H,L}$ and null distributions $\mathcal{Q}_{H,L}$, the status of the hyperedges in L is

always fixed. As fixing the L-indexed inputs can only lower the degree of the distinguishing polynomial p, this assumption holds without loss of generality.

In the null G' is simply uniformly random in $\{\pm 1\}^{\binom{[n]}{r}\setminus\binom{L}{r}}$, i.e., $\mathcal{Q}_{H,L}(G', G_L) = \mathcal{Q}(G')$, where $\mathcal{Q}$ is the uniform distribution. Now, let us focus on G' for the planted $\mathcal{P}_{H,L}$. We can describe the distribution $\mathcal{P}'_{H,L}$ of G' as follows:

1. Choose a random subset S' of $k - \ell$ vertices in $[n] \setminus L$.
2. Choose a random permutation $\pi' \colon S' \to [k] \setminus L$. Extend π' to a permutation from $S' \cup L$ to $[k]$ by setting $\pi'(u) = u$ for all $u \in L$.
3. Set

$$G'(u_1, \ldots, u_r) = \begin{cases} H(\pi'(u_1), \ldots, \pi'(u_r)), & \text{if } u_1, \ldots, u_r \in S' \cup L \\ \text{a random bit}, & \text{otherwise.} \end{cases}$$

Using the above observations we have,

$$\begin{aligned} \mathcal{LR}_D(H, L) &= \sup_{\substack{p \in \mathbb{R}[G(\boldsymbol{u}) : \boldsymbol{u} \in \binom{[n]}{r}] \\ \deg p \le D}} \frac{\mathbf{E}_{\mathcal{P}_{H,L}}[p(G)] - \mathbf{E}_{\mathcal{Q}_{H,L}}[p(G)]}{\sqrt{\mathbf{Var}_{\mathcal{Q}_{H,L}}[p(G)]}} \\ &= \sup_{\substack{p \in \mathbb{R}[G'(\boldsymbol{u}) : \boldsymbol{u} \in \binom{[n]}{r}\setminus\binom{[\ell]}{r}] \\ \deg p \le D}} \frac{\mathbf{E}_{\mathcal{P}'_{H,L}}[p(G')] - \mathbf{E}_{\mathcal{Q}}[p(G')]}{\sqrt{\mathbf{Var}_{\mathcal{Q}}[p(G')]}} \end{aligned}$$

Since the null distribution $\mathcal{Q}$ is a product measure, by a standard linear algebraic argument in the literature of the low-degree method (see [KWB19] or [COGHK+22, Lemma 7.2]), the optimal degree-D polynomial takes an explicit form. Using the expansion with respect to the Fourier-Walsh basis $\{\chi_\alpha(G') = \prod_{e \in \alpha} G'_e, \alpha \subseteq \binom{[n]}{r} \setminus \binom{[\ell]}{r}\}$, the explicit formula for the squared advantage is

$$\mathcal{LR}_D(H, L)^2 = \sum_{\substack{\alpha \subseteq \binom{[n]}{r}\setminus\binom{L}{r} \\ 1 \le |\alpha| \le D}} \widehat{\mathcal{LR}}(\alpha | H, L)^2 \tag{3}$$

where

$$\widehat{\mathcal{LR}}(\alpha | H, L) = \mathbf{E}_{\mathcal{Q}} \frac{\mathcal{P}'_{H,L}(G')}{\mathcal{Q}(G')} \chi_\alpha(G') = \mathbf{E}_{\mathcal{P}'_{H,L}} \chi_\alpha(G').$$

Now we expand the square on the right-hand side of (3) and take the expectation over H.

$$\mathbf{E}_H \mathcal{LR}_D(H, L)^2 = \sum_{\substack{\alpha \subseteq \binom{[n]}{r}\setminus\binom{L}{r} \\ 1 \le |\alpha| \le D}} \mathbf{E} \chi_\alpha(G') \chi_\alpha(G''), \tag{4}$$

where the right-hand expectation is now taken over both the choice of H and the choice of two independent "replicas" G', G'' sampled from $\mathcal{P}'_H$. The joint distribution of G' and G'' is determined by the independent choices of H, the

subsets S', S'', and the permutations π', π''. Equation (4) gives a formula for the second moment of the likelihood ratio with respect to the random variable H, which we spend the rest of this section evaluating; higher moments will be computed later.

We fix $\alpha \subseteq \binom{[n]}{r} \setminus \binom{L}{r}$ and upper bound the expectation. Since we are considering the expectation of a Fourier character, it will often be zero. Let $V(\alpha)$ be the set of vertices in $[n]$ spanned by α. If $S' \cup L$ or $S'' \cup L$ does not entirely contain $V(\alpha)$ then the expectation is zero: if, say, $e \in \alpha'$ is not contained in $S' \cup L$, then $G'(e)$ is independent of all other bits appearing in $\mathbf{E}\chi_\alpha(G')\chi_\alpha(G'') = \prod_{e\in\alpha} G'(e)G''(e)$ resulting in a value of zero. Therefore

$$\begin{aligned}
&\mathbf{E}[\chi_\alpha(G')\chi_\alpha(G'')] \\
&\quad = \mathbf{E}[\chi_\alpha(G')\chi_\alpha(G'') \mid S' \cap S'' \supseteq V(\alpha) \setminus L] \cdot \mathbf{P}[S' \cap S'' \supseteq V(\alpha) \setminus L] \\
&\quad = \mathbf{E}[\chi_\alpha(G')\chi_\alpha(G'') \mid S' \cap S'' \supseteq V(\alpha) \setminus L] \cdot \mathbf{P}[S' \supseteq V(\alpha) \setminus L]^2
\end{aligned} \tag{5}$$

by independence of S' and S''. As S' is a random k-subset of $[n] \setminus L$,

$$\begin{aligned}
\mathbf{P}[S' \supseteq V(\alpha) \setminus L] &= \frac{(k-\ell)(k-\ell-1)\cdots(k-\ell-|V(\alpha)\setminus L|+1)}{(n-\ell)(n-\ell-1)\cdots(n-\ell-|V(\alpha)\setminus L|+1)} \\
&\le \left(\frac{k-\ell}{n-\ell}\right)^{|V(\alpha)\setminus L|}.
\end{aligned} \tag{6}$$

Conditioned on both S' and S'' containing $V(\alpha) \setminus L$,

$$\begin{aligned}
\chi_\alpha(G')\chi_\alpha(G'') &= \prod_{(u_1,\ldots,u_r)\in\alpha} G'(u_1,\ldots,u_r)G''(u_1,\ldots,u_r) \\
&= \prod_{(u_1,\ldots,u_r)\in\alpha} H(\pi'(u_1),\ldots,\pi'(u_r))H(\pi''(u_1),\ldots,\pi''(u_r)).
\end{aligned} \tag{7}$$

As H consists of i.i.d. zero mean ± 1 entries, this expression vanishes in expectation unless every hyperedge in the collection

$$(\psi(u_1),\ldots,\psi(u_r))\colon (u_1,\ldots,u_r) \in \alpha, \psi \in \{\pi', \pi''\}$$

appears exactly twice, in which case the product equals to one. This is only possible if $\pi\colon S' \to S''$ given by $\pi = (\pi'')^{-1} \circ \pi'$ restricts to an automorphism of α. In particular, π must fix the set $V(\alpha)$. As π outside L is a permutation which is chosen uniformly at random, we conclude that (7) is upper bounded by,

$$\begin{aligned}
\mathbf{P}[\pi \text{ fixes } V(\alpha)] &= \frac{|V(\alpha)\setminus L|!}{(k-\ell)(k-\ell-1)\cdots(k-\ell-|V(\alpha)\setminus L|+1)} \\
&\le \left(\frac{|V(\alpha)\setminus L|}{k-\ell}\right)^{|V(\alpha)\setminus L|}.
\end{aligned} \tag{8}$$

Plugging (6) and (8) into (5) and then into (4) yields

$$\mathbf{E}\mathcal{LR}_D(H,L)^2 \le \sum_{\substack{\alpha\subseteq\binom{[n]}{r}\setminus\binom{L}{r} \\ 1\le|\alpha|\le D}} \left(\frac{|V(\alpha)\setminus L|\,(k-\ell)}{(n-\ell)^2}\right)^{|V(\alpha)\setminus L|}. \tag{9}$$

This bound only depends on the hypergraph α through $|V(\alpha) \setminus L|$. For $v = 1, \ldots, rD$ let

$$N(v, D) = \left|\left\{\alpha \subseteq \tbinom{[n]}{r} \setminus \tbinom{L}{r} : |V(\alpha) \setminus L| = v,\ 1 \le |\alpha| \le D\right\}\right|. \tag{10}$$

Grouping the terms on the right-hand side by the value of $v = |V(\alpha) \setminus L|$ gives

$$\mathbf{E}\mathcal{LR}_D(H)^2 \le \sum_{v=1}^{rD} N(v, D) \cdot \left(\frac{v(k-\ell)}{(n-\ell)^2}\right)^v. \tag{11}$$

To finish the proof we will demonstrate that this sum is dominated by the leading term $v = 1$. We split this proof using the following two propositions.

In the first proposition, we bound the "low" vertex size part.

Proposition 1. *Assume that $e(k-\ell)/(n-\ell) \le n^{-\epsilon}$. Then for every $0 < \delta < \epsilon$ it holds for sufficiently large n,*

$$\sum_{v=1}^{\lfloor t \rfloor} N(v, D)\left(\frac{v(k-\ell)}{(n-\ell)^2}\right)^v \le 2^{\binom{\ell}{r-1}} \cdot \frac{n^{-\epsilon}}{1 - n^{-\epsilon+\delta}},$$

where

$$t := e^{-1}(r-1)(\delta \log n)^{1/(r-1)} - \ell \tag{12}$$

In the second proposition, we bound the "high" vertex size part.

Proposition 2. *Assume that $e(k-\ell)/(n-\ell) \le n^{-\epsilon}$. Assume also that for some $\delta > 0$ for which $0 < \delta < \epsilon$, it holds*

1. $\ell \le (r/9)(\delta \log n)^{1/(r-1)}$
   and,
2. $D \le \epsilon\delta^2 (\log n)^{r/(r-1)} / \left(\frac{r}{r-1} \log\log n\right)$.

Then for t given in (12) if also $\delta < 1/4$ it holds,

$$\sum_{v=\lfloor t \rfloor + 1}^{rD} N(v, D)\left(\frac{v(k-\ell)}{(n-\ell)^2}\right)^v \le \exp\left(-\Omega(\delta^{1/(r-1)} \epsilon r (\log n)^{r/(r-1)})\right).$$

Notice now that directly combining both the Propositions for $\delta = \epsilon/4$ directly implies Theorem 1.

3.1 Proof of Proposition 1

Proof. For fixed v, the set $V(\alpha) \setminus L$ can be chosen in $\binom{n-\ell}{v}$ ways. The subset α can then include any of the hyperedges in $V(\alpha)$ of which there are at most $\binom{v+\ell}{r}$,

except those that at completely contained in L of which there are $\binom{\ell}{r}$, leading to the bound:

$$N(v, D) \leq \binom{n-\ell}{v} \cdot 2^{\binom{v+\ell}{r}-\binom{\ell}{r}}. \tag{13}$$

Bounding $N(v, D)$ by (13) and using the standard binomial coefficient bound $\binom{a}{b} \leq (ea/b)^b$, the left hand side is at most

$$\sum_{v=1}^{t} \left(\frac{e(k-\ell)}{n-\ell}\right)^v 2^{\binom{v+\ell}{r}-\binom{\ell}{r}}$$

As $e(k-\ell)/(n-\ell) \leq n^{-\epsilon} = n^{-\epsilon+\delta} \cdot n^{\delta}$, this is bounded by

$$\sum_{v=1}^{t} n^{-(\epsilon-\delta)v} \cdot 2^{-\delta v \log_2 n + \binom{v+\ell}{r}-\binom{\ell}{r}}. \tag{14}$$

Let $f(v) = -\delta v \log_2 n + \binom{v+\ell}{r} - \binom{\ell}{r}, v \geq 1$. For all integer $v \geq 1$,

$$f(v+1) - f(v) = -\delta \log_2 n + \binom{v+\ell}{r-1} \leq -\delta \log n + \left(\frac{e(v+\ell)}{r-1}\right)^{r-1}.$$

By the definition of t, this is negative when $1 \leq v \leq t$, so $f(v)$ is maximized at $v = 1$. Therefore (14) is at most

$$\sum_{v=1}^{\lfloor t \rfloor} n^{-(\epsilon-\delta)v} \cdot 2^{-\delta \log n + \binom{\ell+1}{r}-\binom{\ell}{r}} \leq 2^{\binom{\ell}{r-1}} \cdot \frac{n^{-\epsilon}}{1-n^{-\epsilon+\delta}}$$

using the identity $\binom{\ell+1}{r} - \binom{\ell}{r} = \binom{\ell}{r-1}$ and the geometric sum formula. □

3.2 Proof of Proposition 2

Proof. When v is large, the bound (13) can be improved by taking into account that at most D of the hyperedges can be chosen:

$$\begin{aligned} N(v, D) &\leq \binom{n-\ell}{v} \cdot D^{\binom{\binom{v+\ell}{r}-\binom{\ell}{r}}{D}} \\ &\leq D\left(\frac{e(n-\ell)}{v}\right)^v \cdot \left(\frac{e\binom{v+\ell}{r}}{D}\right)^D \\ &\leq \left(\frac{e(n-\ell)}{v}\right)^v \cdot \left(\frac{e(v+\ell)}{r}\right)^{rD} \cdot D\left(\frac{e}{D}\right)^D. \end{aligned}$$

Under the assumption $e(k-\ell)/(n-\ell) \leq n^{-\epsilon}$ the summation of interest is at most

$$\sum_{v=t+1}^{rD} \left(\frac{e(k-\ell)}{n-\ell}\right)^v \cdot \left(\frac{e(v+\ell)}{r}\right)^{rD} \cdot D\left(\frac{e}{D}\right)^D$$
$$\leq rD^2\left(\frac{e}{D}\right)^D \cdot n^{-\epsilon t}\left(\frac{e(rD+\ell)}{r}\right)^{rD}$$
$$\leq rD^2\left(\frac{e}{D}\right)^D \cdot n^{-\epsilon t}\big(e(D+\ell)\big)^{rD}.$$

As $D \leq \epsilon\delta^2(\log n)^{r/(r-1)}/(\frac{r}{r-1}\log\log n)$ and $\ell \leq (r/9)(\delta \log n)^{1/(r-1)}$, for sufficiently large n,

$$D\log\big((D+\ell)/(\epsilon\delta^2)\big) \leq \epsilon\delta^2(\log n)^{r/(r-1)},$$

Hence, for sufficiently small constant $0 < \delta < 1$, for sufficiently large n it holds

$$D\log\left(e(D+\ell)\right)) \leq \epsilon\delta^2(\log n)^{r/(r-1)},$$

Using also the elementary inequality $D^2(e/D)^D \leq 8$ we conclude that the summation of interest is at most

$$8rn^{-\epsilon t}\exp\left(\epsilon\delta^2(\log n)^{r/(r-1)}\right).$$

Plugging in the direct bound from the definition of t and the upper bound on the leaked vertices,

$$t \geq \frac{r}{7}(\delta\log n)^{1/(r-1)}$$

we conclude that the summation of interest is at most

$$8r\exp\left(-\epsilon\frac{r-1}{e}\delta^{1/(r-1)}(\log n)^{r/(r-1)} + \epsilon\delta^2(\log n)^{r/(r-1)}\right).$$

Choosing now $\delta < 1/4$ concludes the result. □

3.3 Extension to Higher Moments

Now we extend the calculation in Theorem 1 from $p = 1$ to higher p. The $2p$-th moment of $\mathcal{LR}_D(H, L)$ is

$$\mathbf{E}_H\mathcal{LR}_D(H,L)^{2p} = \mathbf{E}_H\left(\sum_{\substack{\alpha\subseteq\binom{[n]}{r}\setminus\binom{L}{r}\\ 1\leq|\alpha|\leq D}} \mathbf{E}_{\substack{G'\sim\mathcal{P}'_H\\ G''\sim\mathcal{P}'_H}}\chi_\alpha(G')\chi_\alpha(G'')\right)^p$$
$$= \sum_{\substack{\alpha_1,\ldots,\alpha_p\subseteq\binom{[n]}{r}\setminus\binom{L}{r}\\ 1\leq|\alpha_i|\leq D}} \mathbf{E}\prod_{i=1}^p \chi_{\alpha_i}(G'_i)\chi_{\alpha_i}(G''_i)$$

where the expectation is over H and also over the replicas G'_i, G''_i sampled independently from $\mathcal{P}'_H$. Each G'_i is equivalently sampled as S'_i and π'_i (and likewise G''_i as S''_i and π''_i).

Fix the Fourier characters $\alpha_1, \ldots, \alpha_p$ and let $V(\alpha_i)$ be the set of vertices in $[n]$ spanned by α_i. First, the expectation is only nonzero if all of the sets S'_i and S''_i contain $V(\alpha_i) \setminus L$. By (6) this occurs with probability at most

$$\mathbf{P}\left[\forall i \in [p].\ S'_i \cap S''_i \supseteq V(\alpha_i) \setminus L\right] \leq \left(\frac{k-\ell}{n-\ell}\right)^{2\sum_{i=1}^p |V(\alpha_i)\setminus L|}. \tag{15}$$

Conditioned on this event,

$$\prod_{i=1}^p \chi_{\alpha_i}(G'_i)\chi_{\alpha_i}(G''_i) = \prod_{i=1}^p \chi_{\pi'_i(\alpha_i)}(H)\chi_{\pi''_i(\alpha_i)}(H)\,.$$

When the expectation is taken over H, this is only nonzero if every hyperedge appears an even number of times among the collection of edges

$$C := (\psi_i(u_1), \ldots, \psi_i(u_r)) : i \in [p],\ (u_1, \ldots, u_r) \in \alpha_i,\ \psi_i \in \{\pi'_i, \pi''_i\}\,.$$

In order for this to occur, every vertex in the image of the ψ_i must be in the image of at least two ψ_i. Let us say that the collection of embeddings is a *double cover* if this occurs. Then

$$\begin{aligned}
&\mathbf{E}_{H,\pi'_i,\pi''_i} \prod_{i=1}^p \chi_{\pi'_i(\alpha_i)}(H)\chi_{\pi''_i(\alpha_i)}(H) \\
&= \mathbf{P}_{\pi'_i,\pi''_i}[C \text{ is an even collection}] \\
&\leq \mathbf{P}_{\pi'_i,\pi''_i}[(\pi'_i, \pi''_i)_{i\in[p]} \text{ is a double cover}]\,.
\end{aligned} \tag{16}$$

Let $V = \sum_{i=1}^p |V(\alpha_i) \setminus L|$. We claim

$$\mathbf{P}_{\pi'_i,\pi''_i}[(\pi'_i, \pi''_i)_{i\in[p]} \text{ is a double cover}] \leq \frac{(2V)^{2V}}{(k-\ell)(k-\ell-1)\cdots(k-\ell-V+1)}\,. \tag{17}$$

This is based on the following surjection a.k.a union bound. The total number of vertices mapped by all the permutations is $2V$. We take any partition of the $2V$ vertices such that every block of the partition has size at least two. There are at most $(2V)^{2V}$ such partitions. We go through the vertices in some fixed order, and for each vertex which is not the first member of its block of the partition, we obtain a factor of $\approx \frac{1}{k-\ell}$ for the probability that the vertex is mapped to the same element as the other members of its block of the partition. Since the blocks have size at least two (in order to be a double cover), we obtain at least V factors of $\approx \frac{1}{k-\ell}$ in this way. We upper bound $\approx \frac{1}{k-\ell}$ by a rising factorial to obtain the bound in (17).

If $V \leq \frac{k-\ell}{2}$, then (17) can simplified to

$$\frac{(2V)^{2V}}{(k-\ell)(k-\ell-1)\cdots(k-\ell-V+1)} \leq \left(\frac{8V^2}{k-\ell}\right)^V. \tag{18}$$

On the other hand, if $V \geq \frac{k-\ell}{2}$, then the right-hand side is at least 1. Combining these two possible cases, we conclude,

$$\mathbf{P}_{\pi'_i,\pi''_i}[(\pi'_i,\pi''_i)_{i\in[p]} \text{ is a double cover}] \leq \left(\frac{8V^2}{k-\ell}\right)^V. \tag{19}$$

Now we return to the main calculation of $\mathbf{E}_H \mathcal{LR}_D(H,L)^{2p}$. Combining (15), (19),

$$\begin{aligned}
\mathbf{E}_H \mathcal{LR}_D(H,L)^{2p} &= \sum_{\substack{\alpha_1,\ldots,\alpha_p \subseteq \binom{[n]}{r}\setminus\binom{L}{r} \\ 1\leq|\alpha_i|\leq D}} \mathbf{E}\prod_{i=1}^{p} \chi_{\alpha_i}(G'_i)\chi_{\alpha_i}(G''_i) \\
&\leq \sum_{\substack{\alpha_1,\ldots,\alpha_p \subseteq \binom{[n]}{r}\setminus\binom{L}{r} \\ 1\leq|\alpha_i|\leq D}} \left(\frac{8V^2(k-\ell)}{(n-\ell)^2}\right)^V \\
&\leq \sum_{\substack{\alpha_1,\ldots,\alpha_p \subseteq \binom{[n]}{r}\setminus\binom{L}{r} \\ 1\leq|\alpha_i|\leq D}} \left(\frac{8p^2D^2(k-\ell)}{(n-\ell)^2}\right)^{\sum_{i=1}^p |V(\alpha_i)\setminus L|} \qquad (V \leq pD) \\
&= \left(\sum_{\substack{\alpha \subseteq \binom{[n]}{r}\setminus\binom{L}{r} \\ 1\leq|\alpha|\leq D}} \left(\frac{8p^2D^2(k-\ell)}{(n-\ell)^2}\right)^{|V(\alpha)\setminus L|}\right)^p
\end{aligned}$$

The inner summation is nearly the combinatorial quantity we bounded in Eq. (9) when computing $\mathbf{E}_H \mathcal{LR}_D(H,L)^2$. The only difference is the factor $8p^2D^2$ which may be larger than what we had before. This factor can be negated by scaling down $\frac{k-\ell}{n-\ell}$. Using the same counting arguments as before with the slightly stronger assumption on k, we conclude the desired moment bound.

4 Cryptographic Applications

4.1 Hypergraph Secret Sharing

The secret sharing scheme of Abram et al. was stated for forbidden graph access structures. The construction extends to partial access structures (R,S) where R is a collection of r-subsets and S consists of all independent sets of R of size at most ℓ.

Construction 2. Forbidden hypergraph secret sharing: Syntactically replace "graph" by "r-uniform hypergraph" and (u,v) by $(u_1,\ldots,u_r)$ in Construction 1.

This scheme reconstructs all $\{u_1,\ldots,u_r\} \in R$ by (2).

Proposition 3. *Assume $(H,\mathcal{P}_{H,L})$ and $(H,\mathcal{Q}_{H,L})$ are (s,ϵ)-indistinguishable for all $L \subseteq V(H)$ with $|L| = \ell$. Then for every independent set $I \subseteq R$ of size at most ℓ, shares of 0 and 1 are $(s,2\epsilon)$-indistinguishable by parties in I.*

Proof. Assume parties in I can 2ϵ-distinguish shares of 0 and 1 using distinguisher D. By the triangle inequality, D ϵ-distinguishes $(H_s, G, \phi(i) : i \in I)$ from $(H, G, \phi(i) : i \in I)$ where

$$G(u_1, \ldots, u_r) = \begin{cases} H(u_1, \ldots, u_r), & \text{if } u_1, \ldots, u_r \in I \\ \text{a random bit}, & \text{otherwise.} \end{cases}$$

for at least one value of s. Let D' be the circuit that, on input $(H', G, u_i : i \in I)$, outputs $D(H' \oplus sR, G, u_i : i \in I)$. As R does not contain any hyperedges within I, by (1), $D'(\mathcal{P}_{H,I})$ is identically distributed to $D(H_s, G, \phi(i) : i \in I)$. As H is random, $D'(\mathcal{Q}_{H,I})$ is identically distributed to $D(H, G, \phi(i) : i \in I)$. Therefore D' and D have the same advantage. □

The class of access structures can be expanded to allow the reconstruction set R to consist of arbitrary sets, as long as the size of all minimal sets is at most r. This is accomplished by a reduction to size exactly r. Let $R' \subseteq [n + r - 1]$ be the r-uniform hypergraph

$$R' = \big\{A \cup \{n+1, \ldots, n+r-|A|\} : A \in R\big\}.$$

Construction 3. Apply Construction 2 to R' with the shares of parties $n + 1, \ldots, n + r - 1$ made public.

If all sets in R' can resconstruct in Construction 2 then all sets in R can reconstruct in Construction 3. As for secrecy, if Construction 2 is secure against all independent sets in R of size at most ℓ, then Construction 3 is secure against such sets of size at most $\ell - r + 1$.

Could Construction 2 give a *provable* separation between the minimum share size of information-theoretic and computational secret sharing? We argue that this is unlikely barring progress in information-theoretic secret sharing lower bounds. The share size in Construction 2 is $(1 + \Omega(1))(\log n)$. However, the share size lower bounds of [KN90, BGK20] do not exceed $\log n$ for any known n-party access structure.

In contrast, Csirmaz [Csi97] proved that there exists an n-party access structure with share size $\Omega(n/\log n)$. Using Csirmaz's method, Beimel [Bei23] constructed *total* r-hypergraph access structures that require share size $\Omega(n^{2-1/(r-1)}/r)$ for every $r \geq 3$.

We argue that Csirmaz's method cannot prove a lower bound exceeding ℓ for any (partial) access structures in which secrecy is required to hold only for sets of size up to ℓ. Csirmaz showed that a scheme with share size s implies the existence of a monotone submodular function f (the joint entropy of the shares in A) from subsets of $\{1, \ldots, n\}$ to real numbers that satisfies the additional constraints

$$f(A) + f(B) \geq f(A \cup B) + f(A \cap B) + 1 \quad \text{if } A, B \in S \text{ and } A \cup B \in R \tag{20}$$

$$f(A) \leq s \quad \text{for all } A \text{ of size 1.} \tag{21}$$

Proposition 4. *Assuming all sets in S have size at most ℓ, there exists a monotone submodular function satisfying (20) and (21) with $s = \ell$.*

As our scheme does not tolerate $\Omega(\log n)$ bits of leakage, the best share size lower bound that can be proved using Csirmaz's relaxation of secret sharing is $\ell = o(\log n)$. The proof of Proposition 4 is a natural generalization of [Csi97, Theorem 3.5] to partial access structures.

Proof (Proposition 4). The function $f(A) = \sum_{t=1}^{|A|} \max\{\ell - t + 1, 0\}$ is monotone, submodular, satisfies (20) for every $R \subseteq \overline{S}$, and (21) with $s = \ell$. □

4.2 Multiparty PSM for Random Functions

Given a function $F\colon [k]^r \to \{\pm 1\}$, the random hypergraph embedding of F is the r-hypergraph $\overline{F}$ on rk vertices $(x, i)\colon x \in [k], i \in [r]$ such that

$$\overline{F}((x_1, 1), \ldots, (x_r, r)) = F(x_1, \ldots, x_r).$$

All other potential hyperedges of $\overline{F}$ are sampled uniformly and independently at random.

We describe the r-partite generalization of Abram et al.'s PSM protocol. Let $\phi\colon [k] \times [r] \to [n]$ be a random injection and let G be the r-hypergraph on n vertices given by

$$G(u_1, \ldots, u_r) = \begin{cases} \overline{F}(\phi^{-1}(u_1), \ldots, \phi^{-1}(u_r)), & \text{if } \phi^{-1}(u_1), \ldots, \phi^{-1}(u_r) \text{ exist} \\ \text{a random bit}, & \text{otherwise.} \end{cases}$$

Construction 4. r-party PSM protocol for F:

In the setup phase, G is published and ϕ is privately given to the parties.
In the evaluation phase,

1. Party i is given input x_i.
2. Party i forwards $u_i = \phi(x_i, i)$ to the evaluator.
3. The evaluator outputs $G(u_1, \ldots, u_r)$.

The protocol is clearly functional. A reasonable notion of security with respect to random functions F should allow the parties' input choices to depend on F. An *input selector* is a randomized function I that, on input F, produces inputs $I(F) = (x_1, \ldots, x_r)$ for the r parties.

We say a protocol is (s', s, ϵ) (simulation) secure against a random function if for every input selector I there exists a size-s' simulator S for which the distributions

$$(F, G, \phi(x_1, 1), \ldots, \phi(x_r, r)) \quad \text{and} \quad (F, S(F, F(x_1, \ldots, x_r))) \tag{22}$$

are (s, ϵ)-indistinguishable, where $(x_1, \ldots, x_r)$ is the output of $I(F)$.

Proposition 5. *Assume* $(H, G, L(H))$ *with* $G \sim \mathcal{P}_{H,L(H)}$ *versus* $G \sim \mathcal{Q}_{H,L(H)}$ *are* (s, ϵ)*-indistinguishable with parameters* $|V(H)| = kr$*,* $|V(G)| = n$*, and* $\ell = r$*. Then Construction 4 is* $(O(\binom{n}{r}), s - O(\binom{n}{r}), \epsilon)$*-secure.*

We label the vertices of H by pairs $(x, r) \in [k] \times [r]$.

Proof. On input (F, y), the simulator S

1. chooses random $u_1, \ldots, u_r \in [n]$
2. sets $G(u_1, \ldots, u_r) = y$
3. samples all other possible hyperedges of G independently at random
4. outputs $(G, u_1, \ldots, u_r)$.

We describe a reduction R that, given a distinguisher D for (22), tells apart $(H, G, L(H))$ with $G \sim \mathcal{P}_{H,L(H)}$ versus $G \sim \mathcal{Q}_{H,L(H)}$ for some leakage function L. On input $(H, G, z_1, \ldots, z_r)$,

1. set F to be the function $F(x_1, \ldots, x_r) = H((x_1, 1), \ldots, (x_r, r))$
2. output $(F, \pi(G), \pi(z_1), \ldots, \pi(z_r))$ for a random permutation π on $[n]$ (which acts on G as a hypergraph isomorphism).

Let L be the leakage function that, on input H, runs $I(F)$ to obtain $(x_1, \ldots, x_r)$, and outputs $((x_1, 1), \ldots, (x_r, r))$.

This reduction preserves distinguishing advantage as it maps the distributions (22) into the distributions $(H, \mathcal{P}_{H,L(H)}, L(H))$ and $(H, \mathcal{Q}_{H,L(H)}, L(H))$, respectively. It can be implemented in size $O(\binom{n}{r})$, giving the desired parameters. □

References

ABF+19. Applebaum, B., Beimel, A., Farràs, O., Nir, O., Peter, N.: Secret-sharing schemes for general and uniform access structures. In: Ishai, Y., Rijmen, V. (eds.) EUROCRYPT 2019. LNCS, vol. 11478, pp. 441–471. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-17659-4_15

ABI+23. Abram, D., Beimel, A., Ishai, Y., Kushilevitz, E., Narayanan, V.: Cryptography from planted graphs: security with logarithmic-size messages. In: Theory of Cryptography Conference, pp. 286–315. Springer, Cham (2023)

AHMS18. Applebaum, B., Holenstein, T., Mishra, M., Shayevitz, O.: The communication complexity of private simultaneous messages, revisited. In: EUROCRYPT (2), pp. 261–286. Springer, Cham (2018)

AKS98. Alon, N., Krivelevich, M., Sudakov, B.: Finding a large hidden clique in a random graph. Rand. Struct. Algorithms **13**(3–4), 457–466 (1998)

BB20. Brennan, M., Bresler, G.: Reducibility and statistical-computational gaps from secret leakage. In: Abernethy, J., Agarwal, S. (eds.) Proceedings of Thirty Third Conference on Learning Theory, Proceedings of Machine Learning Research, vol. 125, pp. 648–847. PMLR (2020)

Bei23. Beimel, A.: Lower bounds for secret-sharing schemes for k-hypergraphs. In: 4th Conference on Information-Theoretic Cryptography (ITC 2023), vol. 267, pp. 16:1–16:13 (2023)

BGK20. Bogdanov, A., Guo, S., Komargodski, I.: Threshold secret sharing requires a linear-size alphabet. Theory Comput. **16**(2), 1–18 (2020)
BHK+19. Barak, B., Hopkins, S., Kelner, J., Kothari, P.K., Moitra, A., Potechin, A.: A nearly tight sum-of-squares lower bound for the planted clique problem. SIAM J. Comput. **48**(2), 687–735 (2019)
CMZ23. Chen, Z., Mossel, E., Zadik, I.: Almost-linear planted cliques elude the Metropolis process. In: Proceedings of the 2023 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 4504–4539. SIAM (2023)
COGHK+22. Coja-Oghlan, A., Gebhard, O., Hahn-Klimroth, M., Wein, A.S., Zadik, I.: Statistical and computational phase transitions in group testing. In: Conference on Learning Theory, pp. 4764–4781. PMLR (2022)
Csi97. Csirmaz, L.: The size of a share must be large. J. Cryptol. **10**(4), 223–231 (1997)
FGR+17. Feldman, V., Grigorescu, E., Reyzin, L., Vempala, S.S., Xiao, Y.: Statistical algorithms and a lower bound for detecting planted cliques. J. ACM **64**(2) (2017)
FK08. Frieze, A., Kannan, R.: A new approach to the planted clique problem. In: IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (2008)
FKN94. Feige, U., Kilian, J., Naor, M.: A minimal model for secure computation (extended abstract). In: Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing (STOC 1994) pp. 554–563. Association for Computing Machinery, New York (1994)
GZ19. Gamarnik , D., Zadik, I.: The landscape of the planted clique problem: dense subgraphs and the overlap gap property. arXiv preprint arXiv:1904.07174 (2019)
Hop18. Hopkins, S.: Statistical Inference and the Sum of Squares Method. Ph.D. thesis, Cornell University (2018)
HS24. Hirahara, S., Shimizu, N.: Planted clique conjectures are equivalent. In: Proceedings of the 56th Annual ACM Symposium on Theory of Computing (STOC 2024), pp. 358–366 (2024)
Hul22. Huleihel, W.: Inferring hidden structures in random graphs. IEEE Trans. Signal Inf. Process. Netw. **8**, 855–867 (2022)
HW21. Holmgren, J., Wein, A.S.: Counterexamples to the low-degree conjecture. In: 12th Innovations in Theoretical Computer Science Conference (ITCS 2021), vol. 185, pp. 75:1–75:9 (2021)
Jer92. Jerrum, M.: Large cliques elude the metropolis process. Rand. Struct. Algorithms **3**, 347–360 (1992)
JP97. Juels, A., Peinado, M.: Hiding cliques for cryptographic security. Des. Codes Crypt. **20**, 11 (1997)
KN90. Kilian, J., Nisan, N.: Unpublished (1990)
Kuc95. Kucera, L.: Expected complexity of graph partitioning problems. Discrete Appl. Math. **57**(2–3), 193–212 (1995)
KWB19. Kunisky, D., Wein, A.S., Bandeira, A.S.: Notes on computational hardness of hypothesis testing: predictions using the low-degree likelihood ratio. In: ISAAC Congress (International Society for Analysis, its Applications and Computation), pp. 1–50. Springer, Cham (2019)
LVW17. Liu, T., Vaikuntanathan, V., Wee, H.: Conditional disclosure of secrets via non-linear reconstruction. In: CRYPTO, pp. 758–790. Springer, Cham (2017)
MNWS+23. Mossel, E., Niles-Weed, J., Sohn, Y., Sun, N., Zadik, I.: Sharp thresholds in inference of planted subgraphs. In: The Thirty Sixth Annual Conference on Learning Theory, pp.)5573–5577. PMLR (2023)

YZZ24. Yu, X., Zadik, I., Zhang, P.: Counting stars is constant-degree optimal for detecting any planted subgraph. arXiv preprint arXiv:2403.17766 (2024)
ZSWB22. Zadik, I., Song, M.J., Wein, A.S., Bruna, J.: Lattice-based methods surpass sum-of-squares in clustering. In: Conference on Learning Theory, pp. 1247–1248. PMLR (2022)

Sparse Linear Regression and Lattice Problems

Aparna Gupte, Neekon Vafa(✉), and Vinod Vaikuntanathan

MIT CSAIL, Cambridge, USA
nvafa@mit.edu

Abstract. Sparse linear regression (SLR) is a well-studied problem in statistics where one is given a design matrix $\mathbf{X} \in \mathbb{R}^{m \times n}$ and a response vector $\mathbf{y} = \mathbf{X}\boldsymbol{\theta}^* + \mathbf{w}$ for a k-sparse vector $\boldsymbol{\theta}^*$ (that is, $\|\boldsymbol{\theta}^*\|_0 \leq k$) and small, arbitrary noise $\mathbf{w}$, and the goal is to find a k-sparse $\widehat{\boldsymbol{\theta}} \in \mathbb{R}^n$ that minimizes the mean squared prediction error $\frac{1}{m}\|\mathbf{X}\widehat{\boldsymbol{\theta}} - \mathbf{X}\boldsymbol{\theta}^*\|_2^2$. While ℓ_1-relaxation methods such as basis pursuit, Lasso, and the Dantzig selector solve SLR when the design matrix is well-conditioned, no general algorithm is known, nor is there any formal evidence of hardness in an *average-case* setting with respect to *all* efficient algorithms.

We give evidence of average-case hardness of SLR w.r.t. all efficient algorithms assuming the worst-case hardness of lattice problems. Specifically, we give an *instance-by-instance* reduction from a variant of the bounded distance decoding (BDD) problem on lattices to SLR, where the condition number of the lattice basis that defines the BDD instance is directly related to the restricted eigenvalue condition of the design matrix, which characterizes some of the classical statistical-computational gaps for sparse linear regression. Also, by appealing to worst-case to average-case reductions from the world of lattices, this shows hardness for a *distribution* of SLR instances; while the design matrices are ill-conditioned, the resulting SLR instances are in the identifiable regime.

Furthermore, for well-conditioned (essentially) *isotropic* Gaussian design matrices, where Lasso is known to behave well in the identifiable regime, we show hardness of outputting *any* good solution in the *unidentifiable* regime where there are many solutions, assuming the worst-case hardness of standard and well-studied lattice problems.

1 Introduction

We study the fundamental statistical problem of *sparse linear regression* where one is given a design matrix $\mathbf{X} \in \mathbb{R}^{m \times n}$ and responses $\mathbf{y} \in \mathbb{R}^m$ where

$$\mathbf{y} = \mathbf{X}\boldsymbol{\theta}^* + \mathbf{w}$$

for a hidden parameter vector $\boldsymbol{\theta}^* \in \mathbb{R}^n$ which is k-sparse, i.e., it has at most k non-zero entries, and a small, arbitrary noise $\mathbf{w} \in \mathbb{R}^m$. The goal is to output a

E. Boyle and M. Mahmoody (Eds.): TCC 2024, LNCS 15365, pp. 276–307, 2025.
https://doi.org/10.1007/978-3-031-78017-2_10

k-sparse vector $\widehat{\boldsymbol{\theta}}$ such that the *(mean squared) prediction error*

$$\frac{1}{m}\|\mathbf{X}\widehat{\boldsymbol{\theta}} - \mathbf{X}\boldsymbol{\theta}^*\|_2^2$$

is as small as possible. Each row of $\mathbf{X}$ corresponds to a *sample* or a *measurement*, and each column of $\mathbf{X}$ corresponds to a *feature* of the model $\boldsymbol{\theta}^*$.

Information-theoretically, it is possible to achieve prediction error

$$\frac{1}{m}\left\|\mathbf{X}\widehat{\boldsymbol{\theta}} - \mathbf{X}\boldsymbol{\theta}^*\right\|_2^2 \leq \frac{4\,\|\mathbf{w}\|_2^2}{m},$$

regardless of the design matrix $\mathbf{X}$. Algorithmically, however, the prediction error achieved by many efficient polynomial-time algorithms (such as ℓ_1-relaxation or ℓ_1-regularization including basis pursuit and Lasso estimators [Tib96, CDS98] as well as the Dantzig selector [CT07]) depends on the conditioning of the design matrix $\mathbf{X}$—specifically, the *restricted-eigenvalue* constant $\zeta(\mathbf{X})$, which essentially lower bounds the smallest singular value of $\mathbf{X}$ restricted to nearly sparse vectors. (See Definition 6 for a formal definition.) By adapting the analysis of [NRWY12] (see Theorems 2 and 3), the thresholded Lasso estimator $\widehat{\boldsymbol{\theta}}_{\mathsf{TL}}$ achieves prediction error[1]

$$\delta_{\mathsf{Lasso}}^2 := \frac{1}{m}\|\mathbf{X}\widehat{\boldsymbol{\theta}}_{\mathsf{TL}} - \mathbf{X}\boldsymbol{\theta}^*\|_2^2 \leq O\left(\frac{\|\mathbf{w}\|_2^2 \cdot k^2}{\zeta(\mathbf{X})^2 \cdot m}\right).$$

In particular, when the design matrix $\mathbf{X}$ is well-conditioned, i.e., $\zeta(\mathbf{X}) = \Omega(1)$ which, for example, happens when the rows are drawn from $\mathcal{N}(\mathbf{0}, \mathbf{I}_{n\times n})$ for $m = \Omega(k \log n)$, the ℓ_1-relaxation or ℓ_1-regularization algorithms achieve the information-theoretically optimal prediction error (up to polynomial factors in k). The smaller the restricted eigenvalue constant, the worse the prediction error bound. We emphasize that without computational bounds, the achievable prediction error does not depend on the characteristics of the design matrix $\mathbf{X}$.

In several naturally occurring high-dimensional regression problems, the design matrix $\mathbf{X}$ may be ill-conditioned as the features, or even the samples, could be correlated. An important question then is to understand *which design matrices admit efficient sparse linear regression algorithms and which do not.* Stated differently, is Lasso (and friends) the best possible algorithm for sparse linear regression? This is the central question of interest in this paper.

A handful of works have started to explore this question both from the algorithmic front and the hardness front. On the algorithmic front, the recent work of [KKMR21] showed an algorithm called *pre-conditioned Lasso* which achieves low prediction error for certain ill-conditioned matrices where Lasso provably fails. On the hardness front, [KKMR22, KKMR21] prove hardness against particular algorithms, namely preconditioned Lasso. In terms of hardness against all efficient algorithms, [ZWJ14] construct a fixed design matrix $\mathbf{X}_{\mathsf{ZWJ}} \in \mathbb{R}^{m\times n}$

[1] This bound assumes $\mathbf{X}$ satisfies a certain column-normalization condition (see Definition 4).

such that solving the k-sparse linear regression problem with better prediction error than Lasso for $k \approx n/4$ on an arbitrary (worst-case) k-sparse ground truth $\boldsymbol{\theta}^*$ implies that $\mathbf{NP} \subseteq \mathbf{P/poly}$.[2] While an exciting initial foray into the landscape of hardness results for sparse linear regression, [ZWJ14] inspires many more questions:

1. Most importantly, which design matrices $\mathbf{X}$ are hard for sparse linear regression? While [ZWJ14] gives us an example in the form of *a single* $\mathbf{X}_{\mathsf{ZWJ}}$, it does not give us much insight into the hardness profile of a given design matrix.
2. The work of [ZWJ14] shows that finding k-sparse solutions for $k \approx n/4$ (i.e., constant factor sparsity) is hard, but is it significantly easier to find much sparser solutions, e.g. what happens with polynomial sparsity, logarithmic sparsity or even constant sparsity? The problem certainly becomes easier, but can we nevertheless show evidence of hardness?
3. A related question is that of fine-grained hardness: while [ZWJ14] shows evidence against polynomial-time algorithms, could there be non-trivial algorithms that solve sparse linear regression significantly faster than brute force search, i.e., in time $n^{o(k)}$?

In this paper, we show hardness results for sparse linear regression that address all of these questions. In a nutshell, and hiding some details, we show an instance-by-instance reduction from (a variant of) the *bounded distance decoding problem* on a lattice with a given basis $\mathbf{B}$ to sparse linear regression w.r.t. a design matrix $\mathbf{X}$ that is essentially drawn from the Gaussian distribution $\mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$, where $\boldsymbol{\Sigma} = \boldsymbol{\Sigma}(\mathbf{B})$ is determined by $\mathbf{B}$. In particular, this allows us to start from conjectured hard instances of lattice problems and construct (nearly) Gaussian design matrices (with a covariance matrix related to the lattice basis) for which k-SLR is hard, partially addressing Question 1. Our main theorem (Theorem 1; see also Remark 2) addresses Questions 2, handling a large range of sparsity parameters k, and 3, by showing a fine-grained reduction.

As a secondary result, even for well-conditioned (essentially) *isotropic* Gaussian design matrices, where Lasso is known to behave well in the identifiable regime, we show hardness of outputting *any* good solution in the less standard *unidentifiable* regime where there are many solutions, assuming the worst-case hardness of standard and well-studied lattice problems.

1.1 Our Results

Binary Bounded Distance Decoding. Our source of hardness is lattice problems. Given a matrix $\mathbf{B} \in \mathbb{R}^{d \times d}$, the lattice generated by $\mathbf{B}$ is

$$\mathcal{L}(\mathbf{B}) = \{\mathbf{Bz} : \mathbf{z} \in \mathbb{Z}^d\}.$$

[2] They show slightly more, i.e., they construct a distribution over $\boldsymbol{\theta}^*$ for which sparse linear regression is hard; however, to the best of our knowledge, this distribution is not polynomial-time sampleable. Indeed, if this distribution were sampleable, their result would show a worst-case to average-case reduction for **NP** which remains a major open problem in complexity theory.

A lattice has many possible bases: indeed, $\mathbf{BU}$ is a basis of $\mathcal{L}(\mathbf{B})$ whenever $\mathbf{U}$ is a unimodular matrix, i.e., an integer matrix with determinant ± 1. The minimum distance $\lambda_1(\mathbf{B})$ of the lattice $\mathcal{L}(\mathbf{B})$ is the (Euclidean) length of the shortest non-zero vector in $\mathcal{L}(\mathbf{B})$. Note that $\lambda_1(\mathbf{B})$ does not depend on $\mathbf{B}$, only on $\mathcal{L}(\mathbf{B})$.[3]

A canonical lattice problem is the bounded distance decoding (BDD): given a basis $\mathbf{B}$ of a lattice, a target vector $\mathbf{t} \in \mathbb{R}^d$, and a parameter $\alpha < 1/2$, find a lattice vector $\mathbf{v} \in \mathcal{L}(\mathbf{B})$ such that $\mathsf{dist}(\mathbf{t}, \mathbf{v}) \leq \alpha \cdot \lambda_1(\mathbf{B})$ given the promise that such a vector exists (equivalently, that $\mathbf{t}$ is sufficiently close to the lattice). By the bound on α, there is a unique such vector. The BDD problem has been very well studied, especially in the last decade, and is widely believed to be hard, for instance, forming the basis of a new generation of post-quantum cryptographic algorithms recently standardized by [NIS]. Our hardness assumption is a variant called *binary BDD* (see, e.g. [KF15]) which asks for a vector $\mathbf{v} \in \mathbf{B} \cdot \{\pm 1\}^d$ that is close to the target $\mathbf{t}$ (as above), again under the promise that such a vector exists. Equivalently, given a vector $\mathbf{t} = \mathbf{Bz} + \mathbf{e}$ where $\mathbf{z} \in \{\pm 1\}^d$ and $\|\mathbf{e}\| \leq \alpha \cdot \lambda_1(\mathbf{B})$, the problem is to find $\mathbf{z}$.

Both BDD and binary BDD are believed to be hard for arbitrary lattice bases $\mathbf{B}$. Indeed, a canonical way to solve (binary as well as regular) BDD is to employ Babai's rounding algorithm [Bab86] which works as follows: compute and output

$$\mathsf{round}(\mathbf{B}^{-1}\mathbf{t}) = \mathsf{round}(\mathbf{z} + \mathbf{B}^{-1}\mathbf{e}) = \mathbf{z} + \mathsf{round}(\mathbf{B}^{-1}\mathbf{e}) \ ,$$

namely, round each coordinate of $\mathbf{B}^{-1}\mathbf{t}$ to the nearest integer. This works as long as each coordinate of $\mathbf{B}^{-1}\mathbf{e}$ is at most $1/2$, which happens as long as $\mathbf{e}$ is small *and* $\mathbf{B}$ has a good condition number. In particular, Babai succeeds if $\alpha = O(1/\kappa(\mathbf{B}))$, where $\kappa(\mathbf{B}) = \sigma_{\max}(\mathbf{B})/\sigma_{\min}(\mathbf{B})$ is the condition number of $\mathbf{B}$. In other words, the condition number of $\mathbf{B}$ determines the performance of the algorithm.

To be sure, there are algorithms for *binary* BDD that perform slightly better than BDD: in particular, [KF15] show a $2^{O(d/\log\log(1/\alpha))}$-time algorithm for binary BDD for a large range of α (see Theorem 7 of the full version [GVV24]). In particular, for $\alpha = 1/\operatorname{poly}(d)$, this becomes a $2^{O(d/\log\log d)}$-time algorithm for binary BDD, whereas the best run-time for BDD algorithms in this regime is $2^{\Theta(d)}$. More than that, [KF15] study binary BDD (and generalizations) in detail and give evidence of hardness: they show reductions from variants of GapSVP and UniqueSVP to (a slight generalization) of binary BDD, as well as a direct reduction from low-density subset sum to binary BDD [KF15, Theorem 14]. In fact, [KF15] use this reduction and their $2^{O(d/\log\log(1/\alpha))}$-time algorithm for binary BDD to give a state-of-the-art algorithm for low-density subset sum. Improv-

[3] We actually use a slightly different definition of $\lambda_1(\mathbf{B})$ which is unimportant for this exposition; we refer the reader to Sect. 2 for more details.

ing on this $2^{O(d/\log\log(1/\alpha))}$ run-time bound for binary BDD would consequently give better algorithms for low-density subset sum.[4]

Our First Result. Our first result shows that for every lattice basis $\mathbf{B}$, there is a related covariance matrix $\boldsymbol{\Sigma} = \boldsymbol{\Sigma}(\mathbf{B})$ such that if binary BDD is hard given the basis $\mathbf{B}$, then k-SLR is hard w.r.t. an *essentially* Gaussian design matrix whose rows are i.i.d. $\mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$.[5] The more precise statement follows. For $\beta \in [0, 2]$, we say that a k-SLR algorithm is a *β-improvement* of Lasso if on input $(\mathbf{X} \in \mathbb{R}^{m\times n}, \mathbf{y} \in \mathbb{R}^m)$, the algorithm achieves prediction error δ^2 where

$$\delta^2 = \delta^2_{\mathsf{Lasso}} \cdot \zeta(\mathbf{X})^\beta \cdot \mathrm{poly}(k, \log n) \le \frac{\|\mathbf{w}\|_2^2}{\zeta(\mathbf{X})^{2-\beta} \cdot m} \cdot \mathrm{poly}(k, \log n).$$

Note that $\beta = 0$ corresponds to Lasso itself, and $\beta = 2$ achieves the information-theoretic bound (up to $\mathrm{poly}(k, \log n)$ factors).

Theorem 1. *There is a $\mathrm{poly}(m, k \cdot 2^{d/k})$-time randomized reduction from* BinaryBDD *in d dimensions with parameter $\alpha \le 1/10$ to k-SLR in dimension $n = k \cdot 2^{d/k}$ and $m \ge 17d$ samples that succeeds with probability $1 - e^{-\Omega(m)}$. Moreover, the reduction maps a* BinaryBDD *instance w.r.t. lattice basis $\mathbf{B}$ to design matrices $\mathbf{X} \in \mathbb{R}^{m\times n}$ where*

- *the distribution of each of the top $m - k$ rows is i.i.d. $\mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$ where $\boldsymbol{\Sigma} = \mathbf{G}_{\mathsf{sparse}}^\top \mathbf{B}^\top \mathbf{B} \mathbf{G}_{\mathsf{sparse}}$ for a fixed, instance-independent matrix $\mathbf{G}_{\mathsf{sparse}} \in \mathbb{R}^{d\times n}$;*
- *each of the bottom k rows is proportional to a fixed, instance-independent vector that depends only on n and k; and*
- *if there is a k-SLR polynomial-time algorithm that is a β-improvement to Lasso, then there is a $\mathrm{poly}(m, k \cdot 2^{d/k})$-time algorithm for* BinaryBDD *in d dimensions with parameter*

$$\alpha \le \frac{1}{\mathrm{poly}(d) \cdot \kappa(\mathbf{B})^{2-\beta}}$$

w.r.t. lattice bases $\mathbf{B}$.

The first qualitative take-away message from our theorem is: if you beat Lasso[6], you beat Babai. The performance of the plain Lasso algorithm is determined by the restricted eigenvalue constant; analogously, the performance of

[4] We remark that the binary LWE (learning with errors) problem, where the LWE secret is binary, is *not* a special case of binary BDD even though LWE is a special case of BDD. Indeed, when writing a binary LWE instance as a BDD instance in the canonical way, only a part of the coefficient vector of the closest lattice point is binary. Thus, even though binary LWE is equivalent in hardness to LWE, [GKPV10, BLP+13, Mic18], we do not know such a statement relating binary BDD and BDD that preserves conditioning of the lattice basis.

[5] The bottom $k \times n$ sub-matrix of $\mathbf{X}$ (call it $\mathbf{X}_2$) is a fixed, worst-case, matrix while each of the top rows is drawn i.i.d. from $\mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$. There are two natural ways to remove the "worst-case" nature of $\mathbf{X}_2$. We discuss them in Sect. 3.1.

[6] More precisely, β-improve Lasso for $\beta > 1$.

Babai's rounding algorithm is bounded by the condition number of the basis matrix. Our theorem says that if you come up with an algorithm that beats the Lasso guarantee, in the sense of achieving prediction error $\frac{1}{\zeta(\mathbf{X})^{2-\beta}}$ for $\beta > 1$, then you have at hand an algorithm that beats the BDD approximation factor achieved by Babai's algorithm by a corresponding amount, namely solve BDD to within a factor of $\frac{1}{\kappa(\mathbf{B})^{2-\beta}}$, whereas Babai's algorithm itself achieves an approximation factor of $\frac{1}{\kappa(\mathbf{B})}$. When $\beta = 2$, the k-SLR algorithm achieves the information-theoretic optimal prediction error (up to polynomial factors in d and k) in which case it gives us a polynomial time binary BDD algorithm for α that is inverse-polynomial in d.

To be sure, there are (recent) algorithms that solve sparse linear regression beating the RE constant bound, e.g. [KKMR21,KKMR23]. These algorithms work by first *pre-conditioning* the design matrix and running the plain Lasso w.r.t. the preconditioned matrix. Our reduction then says that for BDD basis matrices which map to the easy k-SLR instances identified by [KKMR21,KKMR23], you *can* beat Babai. However, that should *not* be surprising, in general: there *are* basis matrices $\mathbf{B}$ which can be "pre-conditioned", e.g. using the Lenstra-Lenstra-Lovász basis reduction algorithm or its variants [LLL82,Sch87], thereby improving their condition number and consequently the performance of Babai. Indeed, in our view, understanding the relationship between the two types of preconditioning transformations—one from the k-SLR world [KKMR21,KKMR23] and the other from the lattice world [LLL82,Sch87]—is a fascinating open question.

On the other hand, there are also basis matrices whose condition number cannot be improved in polynomial time. (If not, then running such a conditioning algorithm and then Babai would give efficient worst-case lattice algorithms, which we believe do not exist.) Indeed, given what we know about the hardness of worst-case lattice problems, our theorem identifies a large class of design matrices where solving sparse linear regression is at least as hard.

Furthermore, our theorem shows the hardness of sparse linear regression for a range of sparsity parameters, addressing Question 2. The sparser the instance, the longer the run-time of the binary BDD algorithm guaranteed by the reduction. On one extreme, when $k = \Omega(d/\log d)$, the reduction runs in $\mathsf{poly}(d)$ time and the sparsity $k = n^{1-\epsilon}$ can be set to any polynomial function of n. On the other extreme, when k is sufficiently super-constant and (say) $\beta = 2$, we get slightly subexponential-time (in d) algorithms for binary BDD, beating the algorithm of [KF15] for $k = \omega(\log\log d)$. Our reduction also addresses Question 3, because gives us fine-grained hardness of k-SLR algorithms running in time $n^{o(k)}$ (see Remark 2).

In order to directly address our Question 1, we can combine our main reduction with known lattice reductions to generate an average-case hard instance for k-SLR. In particular, we can reduce the average-case learning with errors problem [Reg09], to (standard) BDD with bounded norm solutions, and then to binary BDD using a gadget from [MP12] (see full version [GVV24]). In turn, since the learning with errors problem has a reduction from worst-case lattice

problems [Reg09, Pei09], this hardness result relies only on *worst-case* hardness assumptions, despite being *average-case* over k-SLR instances. While the resulting distribution over sparse linear regression design matrices is degenerate and ill-conditioned, the resulting k-SLR instance is still in the identifiable regime and information-theoretically solvable. As far as the authors are aware, this is the first *average-case* hardness result over k-SLR instances, and relies only on *worst-case* lattice assumptions.

Finally, as a teaser to our techniques, we emphasize that lattice problems are fundamentally about integrality, while sparse-linear regression has no integrality constraint. To make our reduction go through, we convert an *integrality* constraint in the lattice problem into a *sparsity* constraint by enforcing an additional linear constraint via a certain gadget matrix. See Sect. 3.2 for more details.

Our Second Result. Our second result, Theorem 6, shows the hardness of sparse linear regression with a design matrix $\mathbf{X}$ whose rows are essentially drawn from the spherical Gaussian $\mathcal{N}(\mathbf{0}, \mathbf{I}_{n\times n})$, as long as the number of samples is smaller than $k \log d$. Even though this puts us in the non-identifiable regime where there may be many $\widehat{\theta}$ consistent with $(\mathbf{X}, \mathbf{y})$, the problem of minimizing the mean squared error[7] is well-defined. More precisely, the top $m - k$ rows of $\mathbf{X}$ are i.i.d. $\mathcal{N}(\mathbf{0}, \mathbf{I}_{n\times n})$ and the bottom k rows are the same fixed matrix from Theorem 1. This is a regime where Lasso would have solved the problem with a larger number of samples, giving some evidence of the optimality of Lasso in terms of sample complexity. The hardness assumption in this case is the continuous learning with errors (CLWE) assumption which was recently shown to be as hard as standard and long-studied worst-case short vector problems on lattices [BRST21, GVV22], which are used as the basis for many post-quantum cryptographic primitives. We state the result formally in Theorem 6 and prove it in Sect. 5.

1.2 Perspectives and Open Problems

We view our results as an initial foray into understanding the landscape of *average-case hardness* of the sparse linear regression problem. One of our results shows a distribution of average-case hard k-SLR instances (under worst-case hardness of lattice problems). However, the design matrices that arise in this distribution are very ill-conditioned; even though the k-SLR instances they define are identifiable, the design matrices are singular and have RE constant 0. An immediate open question arising from this result is whether one can come up with a more robust average-case hard distribution for k-SLR.

More broadly, the connection to lattice problems that we exploit in our reduction inspires a fascinating array of open questions. To begin with, as we briefly discussed above, the recent improvements to k-SLR [KKMR21, KKMR23] proceed via pre-conditioning the design matrix. Analogously, the famed LLL algorithm of [LLL82] from the world of lattices is nothing but a pre-conditioning

[7] In the unidentifiable regime, minimizing the prediction error is not information-theoretically possible without some constraint on the noise.

algorithm for lattice bases. This leads us to ask: is there a constructive way to use (ideas from) lattice basis reduction to get improved k-SLR algorithms? We mention here the results of [GKZ21,ZSWB22] for positive indications of the effectiveness of lattice basis reduction in solving statistical problems. In the lattice world, it is known that if one allows for arbitrary (potentially unbounded-time) pre-processing of a given lattice basis, BDD can be solved with a polynomial approximation factor [LLM06]. Is a similar statement true for k-SLR? We believe an exploration of these questions and others is a fruitful avenue for future research.

Concurrent Work. The concurrent work of [BDT24] shows hardness of achieving non-trivial prediction error in sparse linear regression with Gaussian designs in the improper setting (where the estimator need not be sparse) by reducing from a slight variant of a standard sparse PCA problem. In their setting, they deduce a sample complexity lower bound of (roughly) k^2 for efficient algorithms, where (roughly) k is possible information-theoretically, by assuming a (roughly) k^2 sample complexity lower bound on efficient algorithms for the sparse PCA problem. (Their reduction produces instances in which the noise is standard Gaussian. Our main theorem is written in terms of worst-case noise, but we show a similar statement for independent Gaussian noise in the full version [GVV24].)

1.3 Road Map of Main Results

We prove Theorem 1 through a sequence of steps:

1. In Sect. 3, we state and prove our reduction from $\mathsf{BinaryBDD}_{d,\alpha}$ to k-SLR (Theorem 4).
2. In Sect. 4.2, we give a bound on the restricted-eigenvalue constant ζ of the instances produced by this reduction in terms of $\kappa(\mathbf{B})$, the condition number of the $\mathsf{BinaryBDD}_{d,\alpha}$ lattice instance $\mathbf{B}$ (Lemma 8).
3. In Sect. 4.3, using our reduction and the guarantees of Lasso in terms of ζ (Theorem 3), we state how quantitative *improvements* to Lasso (in terms of the dependence on ζ) give algorithms for $\mathsf{BinaryBDD}_{d,\alpha}$ with quantitatively stronger parameters (Theorem 5).

For the second result (hardness of k-SLR in the non-identifiable regime even for *well-conditioned* design matrices), we give the proof in Sect. 5.

2 Preliminaries

For $n \in \mathbb{N}$, we use the notation $[n] := \{1, 2, \cdots, n\}$. We use the standard definitions of the ℓ_1, ℓ_2, and ℓ_∞ norms in $\mathbb{R}^n$, as well as the ℓ_0 "norm" which is defined as the *sparsity*, or number of non-zero entries of a vector $\mathbf{v} \in \mathbb{R}^n$. We use **bold** notation to denote vectors and matrices. For a matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ and $i \in [m]$, we write $\mathsf{col}_i(\mathbf{A}) \in \mathbb{R}^n$ to denote the ith column of $\mathbf{A}$. We let $I_{n \times n}$ denote the standard identity matrix in $\mathbb{R}^n$. For $\mathbf{v} \in \mathbb{R}^n$, we use the notation $\mathsf{round}(\mathbf{v}) \in \mathbb{Z}^n$

to denote the point-wise rounding function applied to $\mathbf{v} \in \mathbb{R}^n$ (with arbitrary behavior at half integers). We use the standard definitions of the maximum and minimum *singular values* of $\mathbf{A}$ for $m \geq n$:

$$\sigma_{\min}(\mathbf{A}) := \min_{\mathbf{v} \in \mathbb{R}^n} \frac{\|\mathbf{A}\mathbf{v}\|_2}{\|\mathbf{v}\|_2}, \qquad \sigma_{\max}(\mathbf{A}) := \max_{\mathbf{v} \in \mathbb{R}^n} \frac{\|\mathbf{A}\mathbf{v}\|_2}{\|\mathbf{v}\|_2}.$$

We use the notation $\kappa(\mathbf{A})$ to denote the *condition number* of $\mathbf{A}$, defined as

$$\kappa(\mathbf{A}) := \frac{\sigma_{\max}(\mathbf{A})}{\sigma_{\min}(\mathbf{A})}.$$

We use the notation $\mathcal{N}(\mu, \sigma^2)$ to denote the univariate normal distribution with mean μ and standard deviation σ. Similarly, we use the notation $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ to denote the multivariate normal distribution with mean $\boldsymbol{\mu} \in \mathbb{R}^n$ and positive semi-definite covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{n \times n}$. We emphasize that we do not require $\boldsymbol{\Sigma}$ to be positive definite.

2.1 Bounded Distance Decoding

We now define lattice quantities that will be useful for us.

Definition 1. *For a full-rank matrix $\mathbf{B} \in \mathbb{R}^{d \times d}$, the lattice $\mathcal{L}(\mathbf{B})$ generated by basis $\mathbf{B}$ consists of the set all integer linear combinations of the columns of $\mathbf{B}$. As is standard, we define*

$$\lambda_1(\mathbf{B}) := \min_{\mathbf{z} \in \mathbb{Z}^d \setminus \{\mathbf{0}\}} \|\mathbf{B}\mathbf{z}\|_2 ,$$

which corresponds to the shortest distance between any two elements of $\mathcal{L}(\mathbf{B})$. In this paper, since we consider a (± 1) binary *version of the bounded distance decoding problem, we define the quantity*

$$\lambda_{1,\mathsf{bin}}(\mathbf{B}) := \min_{\mathbf{z}_1 \neq \mathbf{z}_2 \in \{1,-1\}^d} \|\mathbf{B}\mathbf{z}_1 - \mathbf{B}\mathbf{z}_2\|_2 .$$

Note that while $\lambda_1(\mathbf{B})$ is a basis-independent quantity for the lattice $\mathcal{L}(\mathbf{B})$, $\lambda_{1,\mathsf{bin}}(\mathbf{B})$ depends on the basis $\mathbf{B}$.

Lemma 1. *For any matrix $\mathbf{B} \in \mathbb{R}^{d \times d}$, we have $\sigma_{\min}(\mathbf{B}) \leq \lambda_1(\mathbf{B}) \leq \lambda_{1,\mathsf{bin}}(\mathbf{B}) \leq 2\sigma_{\max}(\mathbf{B})$.*

Proof. It is immediate that $\lambda_1(\mathbf{B}) \leq \lambda_{1,\mathsf{bin}}(\mathbf{B})$. Let $\mathbf{e} \in \mathbb{Z}^d$ be any standard basis vector. We have

$$\lambda_{1,\mathsf{bin}}(\mathbf{B}) = \min_{\mathbf{z}_1 \neq \mathbf{z}_2 \in \{1,-1\}^d} \|\mathbf{B}(\mathbf{z}_1 - \mathbf{z}_2)\|_2 \leq 2\,\|\mathbf{B}\mathbf{e}\|_2 \leq 2\sigma_{\max}(\mathbf{B})\,\|\mathbf{e}\|_2 = 2\sigma_{\max}(\mathbf{B}).$$

We also have

$$\lambda_1(\mathbf{B}) = \min_{\mathbf{z} \in \mathbb{Z}^d \setminus \{\mathbf{0}\}} \|\mathbf{B}\mathbf{z}\|_2 \geq \min_{\mathbf{z} \in \mathbb{Z}^d \setminus \{\mathbf{0}\}} \sigma_{\min}(\mathbf{B})\,\|\mathbf{z}\|_2 \geq \sigma_{\min}(\mathbf{B}),$$

since all $\mathbf{z} \in \mathbb{Z}^d \setminus \{\mathbf{0}\}$ satisfy $\|\mathbf{z}\|_2 \geq 1$.

Definition 2 ($\mathsf{BinaryBDD}_{d,\alpha}$)**.** *Let* $\mathsf{BinaryBDD}_{d,\alpha}$ *be the following worst-case search problem in dimension* $d \in \mathbb{N}$ *with parameter* $\alpha < 1/2$*. Given full rank* $\mathbf{B} \in \mathbb{R}^{d\times d}$ *and* $\mathbf{t} \in \mathbb{R}^d$*, output* $\mathbf{z} \in \{-1,1\}^d$ *such that* $\|\mathbf{Bz}-\mathbf{t}\|_2 \le \alpha \cdot \lambda_{1,\mathsf{bin}}(\mathbf{B})$ *(if one exists). Equivalently, given* $(\mathbf{B}, \mathbf{Bz}+\mathbf{e})$ *for* $\|\mathbf{e}\|_2 \le \alpha \cdot \lambda_{1,\mathsf{bin}}(\mathbf{B})$*, output* $\mathbf{z} \in \{-1,1\}^d$*.*

Note that by definition of $\lambda_{1,\mathsf{bin}}(\mathbf{B})$, the constraint $\alpha < 1/2$ guarantees the uniqueness of $\mathbf{z} \in \{-1,1\}^d$. As an aside, the standard BDD problem instead uses $\lambda_1(\mathbf{B})$ and allows outputting any $\mathbf{z} \in \mathbb{Z}^d$ instead of $\mathbf{z} \in \{-1,1\}^d$. Additionally, note that we could easily consider full-rank $\mathbf{B} \in \mathbb{R}^{d_1\times d}$ for $d \le d_1 \le O(d)$ and our main results would all go through, and similarly for larger d_1 with a slight change in the parameter dependence. Furthermore, even though we write $\mathbf{B} \in \mathbb{R}^{d\times d}$, we will assume the entries of $\mathbf{B}$ have $\mathrm{poly}(d)$ bits of precision. (This is just for simplicity and convenience; one can modify the corresponding algorithmic statements so that they run in polynomial time in the description length of the basis $\mathbf{B}$.)

We note that there is a reduction from general BDD to BinaryBDD, at the cost of making the instance degenerate (in the sense of having a non-trivial kernel). See the full version [GVV24].

We now recall standard spectral bounds for random Gaussian matrices.

Lemma 2 (As in [RV10]). *Let* $\mathbf{R} \in \mathbb{R}^{m\times d}$ *be such that* $R_{i,j} \sim_{\text{i.i.d.}} \mathcal{N}(0,1)$*. For all* $t > 0$*, we have*

$$\Pr\left[\sqrt{m}-\sqrt{d}-t \le \sigma_{\min}(\mathbf{R}) \le \sigma_{\max}(\mathbf{R}) \le \sqrt{m}+\sqrt{d}+t\right] \ge 1-2e^{-t^2/2}.$$

In particular, for $m \ge 16d$*, by setting* $t = \sqrt{m}/4$*, we have*

$$\Pr\left[\frac{\sqrt{m}}{2} \le \sigma_{\min}(\mathbf{R}) \le \sigma_{\max}(\mathbf{R}) \le \frac{3\sqrt{m}}{2}\right] \ge 1-2e^{-m/32}.$$

Throughout the paper, let $\chi^2(m)$ denote the chi-squared distribution with m degrees of freedom.

Lemma 3 (As in [LM00], Corollary of Lemma 1). *Let* $X \sim \chi^2(m)$*, i.e., X is distributed according to the* χ^2 *distribution with m degrees of freedom. For any* $t \ge 0$*, we have*

$$\Pr[X \ge m+2\sqrt{tm}+2t] \le e^{-t}.$$

In particular, setting $t = m/4$*, we get*

$$\Pr[X \ge 5m/2] \le e^{-m/4}.$$

2.2 Sparse Linear Regression

We now define the problem of k-sparse linear regression (k-SLR for short), which, for computational complexity simplicity, is phrased in terms of mean squared error (as opposed to mean squared prediction error).

Definition 3 (k-SLR)**.** *Given a design matrix* $\mathbf{X} \in \mathbb{R}^{m\times n}$*, a target vector* $\mathbf{y} \in \mathbb{R}^m$*, and* $\delta > 0$*, output a* k*-sparse* $\widehat{\boldsymbol{\theta}} \in \mathbb{R}^n$ *such that*

$$\frac{\|\mathbf{X}\widehat{\boldsymbol{\theta}} - \mathbf{y}\|_2^2}{m} \leq \delta^2,$$

assuming one exists.

We follow the convention of [NRWY12] of working with design matrices $\mathbf{X}$ that are *column-normalized*:

Definition 4 (As in [NRWY12]). *We say a matrix* $\widetilde{\mathbf{X}} \in \mathbb{R}^{m\times n}$ *is* column-normalized *if for all* $i \in [n]$*,*

$$\left\|\mathsf{col}_i(\widetilde{\mathbf{X}})\right\|_2 \leq \sqrt{m}.$$

Now, we show that a column-normalized $\widetilde{\mathbf{X}}$ also satisfies a spectral bound with respect to sparse vectors.

Lemma 4. *Suppose* $\widetilde{\mathbf{X}} \in \mathbb{R}^{m\times n}$ *is column-normalized. Then,*

$$\max_{\substack{\|\mathbf{v}\|_2=1,\\ \|\mathbf{v}\|_0\leq k}} \left\|\widetilde{\mathbf{X}}\mathbf{v}\right\|_2 \leq \sqrt{km}.$$

Proof. Let $\mathbf{v}$ be a vector such that $\|\mathbf{v}\|_2 = 1$ and $\|\mathbf{v}\|_0 \leq k$. By the triangle inequality and column normalization, we have

$$\left\|\widetilde{\mathbf{X}}\mathbf{v}\right\|_2 = \left\|\sum_{i\in[n]} v_i \cdot \mathsf{col}_i(\widetilde{\mathbf{X}})\right\|_2 \leq \sum_{i\in[n]} |v_i| \cdot \left\|\mathsf{col}_i(\widetilde{\mathbf{X}})\right\|_2 \leq \sqrt{m} \cdot \|\mathbf{v}\|_1 \leq \sqrt{km},$$

where the last inequality holds because $\|\mathbf{v}\|_1 \leq \sqrt{k}\,\|\mathbf{v}\|_2 = \sqrt{k}$.

To analyze the performance Lasso algorithm on the instances of k-SLR we obtain from our reduction, we slightly modify the analysis of [NRWY12] to allow for a more flexible definition of the *restricted eigenvalue* (RE) constant of a matrix. First, we define the following cone.

Definition 5. *Let* $S \subseteq [n]$ *and* $\epsilon > 0$*. We define* $\mathbb{C}_\epsilon(S)$*, the* ϵ*-cone for* $S \subseteq [n]$*, as follows:*

$$\mathbb{C}_\epsilon(S) := \{\Delta \in \mathbb{R}^n : \|\Delta_{\bar{S}}\|_1 \leq (1+\epsilon)\|\Delta_S\|_1\}.$$

The notation $\bar{S}$ *denotes the complement of* S*, namely* $[n] \setminus S$*, and the notation* $\Delta_I \in \mathbb{R}^{|I|}$ *denotes the restriction of* Δ *to coordinates in* $I \subseteq [n]$*.*

A standard definition (e.g., as in [NRWY12]) sets $\epsilon = 2$, but we will use the flexibility of setting ϵ close to 0. Now, we define the restricted eigenvalue condition.

Definition 6. *Suppose* $\widetilde{\mathbf{X}} \in \mathbb{R}^{m \times n}$ *is column-normalized. Then, we say* $\widetilde{\mathbf{X}}$ *satisfies the* (ϵ, ζ)-restricted eigenvalue (RE) condition *for* $S \subseteq [n]$ *if*

$$\frac{\left\|\widetilde{\mathbf{X}}\boldsymbol{\theta}\right\|_2^2}{m \cdot \|\boldsymbol{\theta}\|_2^2} \geq \zeta$$

for all $\boldsymbol{\theta} \in \mathbb{C}_\epsilon(S)$.

Note that this restricted eigenvalue condition corresponds to a restricted form of a strong convexity condition over the loss function

$$\mathcal{L}(\boldsymbol{\theta}) := \frac{1}{2m}\left\|\mathbf{y} - \widetilde{\mathbf{X}}\boldsymbol{\theta}\right\|_2^2.$$

Also note that the ℓ_1-regularization-based Lasso estimator does not output a k-sparse solution. To obtain a k-sparse solution, we truncate $\widehat{\boldsymbol{\theta}}_\lambda$ to the k entries of largest absolute value, to obtain the *thresholded Lasso* estimate $\widehat{\boldsymbol{\theta}}_{\mathsf{TL}}$. Lemma 9 of [ZWJ14] shows that $\left\|\widehat{\boldsymbol{\theta}}_{\mathsf{TL}} - \boldsymbol{\theta}^*\right\|_2 \leq 5\left\|\widehat{\boldsymbol{\theta}}_\lambda - \boldsymbol{\theta}^*\right\|_2$.

Now, we state a modified version of Corollary 1 of [NRWY12], which tells us an error bound on the thresholded Lasso estimator, given our modified definition of the restricted eigenvalue condition.

Theorem 2 (Modified Version of Corollary 1 of [NRWY12]). *Let* $\epsilon \in (0, 2]$, *let* $\widetilde{\mathbf{X}} \in \mathbb{R}^{m \times n}$ *be column-normalized, and let* $\widetilde{\mathbf{y}} = \widetilde{\mathbf{X}}\boldsymbol{\theta}^* + \widetilde{\mathbf{w}}$ *for some* k*-sparse vector* $\boldsymbol{\theta}^*$ *supported on* S *for* $S \subseteq [n], |S| = k$. *Suppose* $\widetilde{\mathbf{X}}$ *satisfies the* (ϵ, ζ)*-restricted eigenvalue condition for* S. *As long as*

$$\lambda \geq \frac{2+\epsilon}{\epsilon} \cdot \left\|\frac{1}{m}\widetilde{\mathbf{X}}^\top \widetilde{\mathbf{w}}\right\|_\infty,$$

and $\lambda > 0$, *then any Lasso solution* $\widehat{\boldsymbol{\theta}}_\lambda$ *with regularization parameter* λ *satisfies the bound*

$$\left\|\widehat{\boldsymbol{\theta}}_\lambda - \boldsymbol{\theta}^*\right\|_2^2 \leq O\left(\frac{\lambda^2 k}{\zeta^2}\right).$$

While the bounds in the theorem statement above do not depend on column normalization of $\widetilde{\mathbf{X}}$ (or corresponding normalization of $\widetilde{\mathbf{w}}$), our definition of the restricted eigenvalue condition needs the matrix to be column-normalized (in particular, so that the RE constant is scale invariant with respect to $\mathbf{B}$). The ϵ dependence in λ comes from modifying [NRWY12, Lemma 1], which needs the bound on λ as above to guarantee that the optimal error $\widehat{\Delta} := \widehat{\boldsymbol{\theta}}_\lambda - \boldsymbol{\theta}^*$ satisfies $\widehat{\Delta} \in \mathbb{C}_\epsilon(S)$.

Theorem 3. *Let* $\epsilon \in (0, 2]$, *let* $\widetilde{\mathbf{X}} \in \mathbb{R}^{m \times n}$ *be column-normalized, and let* $\widetilde{\mathbf{y}} = \widetilde{\mathbf{X}}\boldsymbol{\theta}^* + \widetilde{\mathbf{w}}$ *for some* k*-sparse vector* $\boldsymbol{\theta}^*$ *supported on* S *for* $S \subseteq [n], |S| = k$.

Suppose $\widetilde{\mathbf{X}}$ *satisfies the* (ϵ, ζ)*-restricted eigenvalue condition for* S*. For an optimally chosen positive regularization parameter, the thresholded Lasso solution* $\widehat{\boldsymbol{\theta}}_{\mathsf{TL}}$ *with satisfies the prediction error bound*

$$\frac{1}{m}\|\widetilde{\mathbf{X}}\widehat{\boldsymbol{\theta}}_{\mathsf{TL}} - \widetilde{\mathbf{X}}\boldsymbol{\theta}^*\|_2^2 \leq O\left(\frac{\|\widetilde{\mathbf{w}}\|_2^2 \cdot k^2}{\zeta^2 \cdot \epsilon^2 \cdot m}\right).$$

Proof. From Theorem 2, we know that for

$$\lambda \geq \frac{2+\epsilon}{\epsilon} \cdot \left\|\frac{1}{m}\widetilde{\mathbf{X}}^\top \widetilde{\mathbf{w}}\right\|_\infty,$$

we have

$$\left\|\widehat{\boldsymbol{\theta}}_\lambda - \boldsymbol{\theta}^*\right\|_2 \leq O\left(\frac{\lambda\sqrt{k}}{\zeta}\right).$$

By [ZWJ14, Lemma 9], we therefore have

$$\left\|\widehat{\boldsymbol{\theta}}_{\mathsf{TL}} - \boldsymbol{\theta}^*\right\|_2 \leq O\left(\frac{\lambda\sqrt{k}}{\zeta}\right).$$

Plugging in the optimal choice of λ yields

$$\left\|\widehat{\boldsymbol{\theta}}_{\mathsf{TL}} - \boldsymbol{\theta}^*\right\|_2 \leq O\left(\frac{\sqrt{k}}{\epsilon \cdot \zeta} \cdot \left\|\frac{1}{m}\widetilde{\mathbf{X}}^\top \widetilde{\mathbf{w}}\right\|_\infty\right).$$

Since $\widetilde{\mathbf{X}}$ is column-normalized, we know

$$\left\|\frac{1}{m}\widetilde{\mathbf{X}}^\top \widetilde{\mathbf{w}}\right\|_\infty = \frac{1}{m}\max_{i\in[n]}\left|\left\langle \mathsf{col}_i(\widetilde{\mathbf{X}}), \widetilde{\mathbf{w}}\right\rangle\right| \leq \frac{1}{m}\left\|\mathsf{col}_i(\widetilde{\mathbf{X}})\right\|_2 \|\widetilde{\mathbf{w}}\|_2 \leq \frac{\|\widetilde{\mathbf{w}}\|_2}{\sqrt{m}},$$

where the first inequality is due to Cauchy-Schwarz, and the second inequality is due to column normalization. Combining the two above inequalities, we have

$$\left\|\widehat{\boldsymbol{\theta}}_{\mathsf{TL}} - \boldsymbol{\theta}^*\right\|_2 \leq O\left(\frac{\sqrt{k} \cdot \|\widetilde{\mathbf{w}}\|_2}{\epsilon \cdot \zeta \cdot \sqrt{m}}\right).$$

Since $\widehat{\boldsymbol{\theta}}_{\mathsf{TL}} - \boldsymbol{\theta}^*$ is $2k$-sparse, we can apply Lemma 4 to get

$$\left\|\widetilde{\mathbf{X}}\widehat{\boldsymbol{\theta}}_{\mathsf{TL}} - \widetilde{\mathbf{X}}\boldsymbol{\theta}^*\right\|_2 \leq O\left(\frac{k \cdot \|\widetilde{\mathbf{w}}\|_2}{\epsilon \cdot \zeta}\right).$$

Squaring and dividing by m gives the desired result.

Remark 1. We remark that Theorems 2 and 3 imply that in the noiseless setting of k-SLR, that is, when $\mathbf{y} = \mathbf{X}\boldsymbol{\theta}^*$, Lasso not only achieves prediction error 0 but also recovers the ground truth $\boldsymbol{\theta}^*$ as long as the restricted eigenvalue constant is strictly positive. In particular, this means that any reduction showing the **NP**-hardness of noiseless sparse linear regression must produce instances with design matrices with restricted eigenvalue 0 (unless $\mathbf{P} = \mathbf{NP}$).

However, there is no reason information theoretically that the prediction error should have dependence on the RE constant ζ. We now state a bound on the prediction error of the optimal ℓ_0 predictor, which need not be computationally efficient.

Proposition 1. *Let* $\mathbf{X} \in \mathbb{R}^{m \times n}$*, and let* $\mathbf{y} = \mathbf{X}\boldsymbol{\theta}^* + \mathbf{w}$ *for some* k*-sparse* $\boldsymbol{\theta}^* \in \mathbb{R}^n$*. The* ℓ_0*-predictor*

$$\widehat{\boldsymbol{\theta}} \in \operatorname*{arg\,min}_{\|\boldsymbol{\theta}\|_0 \le k} \|\mathbf{X}\boldsymbol{\theta} - \mathbf{y}\|_2^2$$

satisfies the prediction error bound

$$\frac{\left\|\mathbf{X}\widehat{\boldsymbol{\theta}} - \mathbf{X}\boldsymbol{\theta}^*\right\|_2^2}{m} \le \frac{4\|\mathbf{w}\|_2^2}{m}.$$

Proof. By definition of $\widehat{\boldsymbol{\theta}}$ and since $\boldsymbol{\theta}^*$ is k-sparse, we know

$$\left\|\mathbf{X}\widehat{\boldsymbol{\theta}} - \mathbf{y}\right\|_2 \le \|\mathbf{X}\boldsymbol{\theta}^* - \mathbf{y}\|_2 = \|\mathbf{w}\|_2 .$$

By the triangle inequality,

$$\left\|\mathbf{X}\widehat{\boldsymbol{\theta}} - \mathbf{X}\boldsymbol{\theta}^*\right\|_2 \le \left\|\mathbf{X}\widehat{\boldsymbol{\theta}} - \mathbf{y}\right\|_2 + \|\mathbf{y} - \mathbf{X}\boldsymbol{\theta}^*\|_2 \le 2\|\mathbf{w}\|_2 .$$

The bound immediately follows by squaring and dividing by m.

Comparing Theorem 3 and Proposition 1 highlights a critical computational-statistical gap: the prediction error for the Lasso algorithm depends quantitatively on the restricted eigenvalue condition of the design matrix, whereas the information-theoretic minimizer has prediction error that is completely independent of the restricted eigenvalue condition.

3 Reduction from Bounded Distance Decoding

Our main result in this section is a reduction from $\mathsf{BinaryBDD}_{d,\alpha}$ to k-SLR:

Theorem 4. *Let* d, m, n, k *be integers such that* k *divides* d*,* $m \ge 17d$ *and* $n = k \cdot 2^{d/k}$*. Then, there is a* $\mathrm{poly}(n, m)$*-time reduction from* $\mathsf{BinaryBDD}_{d,\alpha}$ *in dimension* d *with parameter* $\alpha \le 1/10$ *to* k-SLR *in dimension* n *with* m *samples. The reduction is randomized and succeeds with probability* $1 - e^{-\Omega(m)}$*. Moreover, the reduction maps a* $\mathsf{BinaryBDD}_{d,\alpha}$ *instance* $(\mathbf{B}, \cdot)$ *to* k-SLR *instances with design matrices* $\mathbf{X} \in \mathbb{R}^{m \times n}$ *and parameter* $\delta = \Theta(\lambda_{1,\mathsf{bin}}(\mathbf{B}))$ *such that* $\mathbf{X}$ *that can be decomposed as*

$$\mathbf{X} = \begin{pmatrix} \mathbf{X}_1 \\ \mathbf{X}_2 \end{pmatrix},$$

where the distribution of each row of $\mathbf{X}_1 \in \mathbb{R}^{(m-k) \times n}$ *is distributed as* i.i.d. $\mathcal{N}(\mathbf{0}, \mathbf{G}_{\mathsf{sparse}}^\top \mathbf{B}^\top \mathbf{B} \mathbf{G}_{\mathsf{sparse}})$ *for a fixed, instance-independent matrix* $\mathbf{G}_{\mathsf{sparse}} \in \mathbb{R}^{d \times n}$*, and* $\mathbf{X}_2 \in \mathbb{R}^{k \times n}$ *is proportional to a fixed, instance-independent matrix that depends only on* n *and* k*.*

Remark 2. This reduction also gives us a *fine-grained* hardness result. Suppose there is an algorithm for k-SLR in n dimensions with m samples that runs in time $\mathrm{poly}(m,k)\cdot(n/k)^{k^{1-\epsilon}}$. Then, the above reduction gives an algorithm for $\mathsf{BinaryBDD}_{d,\alpha}$ in d dimensions that runs in time $\mathrm{poly}(d)\cdot 2^{d/k^{\epsilon}}$.

[GV21] also prove a fine-grained hardness result for k-SLR from a lattice problem, specifically the closest vector problem. However, the key difference is that their result is in the unidentifiable regime, whereas ours is in the identifiable regime, i.e. the k-SLR instances produced by the reduction from binary BDD have a unique solution. Further, their fine-grained hardness results are for worst-case covariates, whereas our hardness results produces covariates drawn from a Gaussian (with a covariance matrix determined by the binary-BDD basis).

3.1 Interpretations

We now interpret the distribution of the design matrices $\mathbf{X}$ in the reduction in Theorem 4. As stated, the sub-matrix $\mathbf{X}_2$ is phrased as a worst-case design matrix, while $\mathbf{X}_1$ has i.i.d. $\mathcal{N}(\mathbf{0}, \mathbf{G}_{\mathsf{sparse}}^{\top}\mathbf{B}^{\top}\mathbf{B}\mathbf{G}_{\mathsf{sparse}})$ rows, where the covariance matrix is tightly related to the $\mathsf{BinaryBDD}_{d,\alpha}$ instance $\mathbf{B}$. (The definition of the fixed, instance-independent gadget matrix $\mathbf{G}_{\mathsf{sparse}}$ is given in Sect. 3.2.)

There are two natural ways to remove the "worst-case" nature of $\mathbf{X}_2$ in our instances:

1. Our reduction still holds if the $\mathbf{X}_2$ part of our design matrix is removed, and instead, one enforced an exact linear constraint on $\boldsymbol{\theta}$ (specifically, $\mathbf{G}_{\mathsf{partite}}\boldsymbol{\theta} = \mathbf{1}$, where $\mathbf{G}_{\mathsf{partite}}$ is defined in Sect. 3.2). This corresponds to constraining the k-sparse regression vector $\boldsymbol{\theta} \in \mathbb{R}^n$ to an affine subspace. Since many of the techniques for sparse linear regression involve convex optimization, intersecting the solution landscape with this affine subspace will preserve convexity, so these algorithms would still work if the design matrix were just $\mathbf{X}_1$ on its own, without the worst-case $\mathbf{X}_2$ sub-matrix.
2. Alternatively, one can interpret each row of $\mathbf{X}_2$ as a mean of a Gaussian distribution with a covariance matrix $\sigma^2 I_{n\times n}$ for very small $\sigma > 0$. By reordering the rows of $\mathbf{X}$ and entries of the target $\mathbf{y}$, all rows of $\mathbf{X}$ can now be drawn i.i.d. from a *mixture* of $k+1$ Gaussians, with weight $(m-k)/m$ on $\mathcal{N}(\mathbf{0}, \mathbf{G}_{\mathsf{sparse}}^{\top}\mathbf{B}^{\top}\mathbf{B}\mathbf{G}_{\mathsf{sparse}})$ and weight $1/m$ on $\mathcal{N}(\mathbf{v}_i, \sigma^2 I_{n\times n})$ for all $i \in [k]$, where $\mathbf{v}_i$ is the ith row of $\mathbf{X}_2$ and $\sigma > 0$ is very small. Our reduction will go through (with modified parameters) as long as we increase the number of samples m to roughly $O(m\log(k))$ so that each row of $\mathbf{X}_2$ is hit by a coupon collector argument.

3.2 Proof of Theorem 4

Before proving Theorem 4, we introduce some notation and gadgets that will be useful for us. For $n, k \in \mathbb{N}$ where n is a multiple of k, let $\mathcal{S}_{n,k}$ denote the set of k-sparse, k-partite binary vectors $\mathbf{v} \in \{0,1\}^n$. That is, for each $i \in \{0, \cdots, k-1\}$,

there exists a unique $j \in [n/k]$ such that $v_{in/k+j} = 1$, and all other entries of $\mathbf{v}$ are 0.

Let $\mathbf{G}_{\mathsf{sparse}} \in \mathbb{R}^{d \times n}$ be the following (rectangular) block diagonal matrix, which will allow us to map binary vectors to binary sparse partite vectors. Each of the k blocks $\mathbf{H} \in \mathbb{R}^{d/k \times n/k}$ are identical, with $n/k = 2^{d/k}$, and has columns consisting of all vectors $\{-1, 1\}^{d/k}$, in some fixed order.

Lemma 5. *The matrix* $\mathbf{G}_{\mathsf{sparse}}$ *invertibly maps* $\mathcal{S}_{n,k}$ *to* $\{-1, 1\}^d$.

Proof. Observe that $|\mathcal{S}_{n,k}| = (n/k)^k = 2^d = |\{-1, 1\}^d|$, so it suffices to show that $\mathbf{G}_{\mathsf{sparse}}$ is surjective. Let $\mathbf{z} \in \{-1, 1\}^d$. Breaking $\mathbf{z}$ up into k contiguous blocks of d/k coordinates, we can identify each block of $\mathbf{z}$ in $\{-1, 1\}^{d/k}$ with a unique column in $\mathbf{H}$. Let $\mathbf{v} \in \mathcal{S}_{n,k}$ be defined so that for the ith block, $v_{ik+j} = 1$ if and only if the jth column of $\mathbf{H}$ is equal to the ith part of $\mathbf{z}$. It then follows that $\mathbf{G}_{\mathsf{sparse}}\mathbf{v} = \mathbf{z}$, as desired.

We now define a different gadget matrix, $\mathbf{G}_{\mathsf{partite}} \in \mathbb{R}^{k \times n}$, that will help enforce the regression vector to be partite and binary. The matrix $\mathbf{G}_{\mathsf{partite}}$ is once again block diagonal with k identical blocks, where each block is now the vector $\mathbf{1}^\top \in \mathbb{R}^{1 \times n/k}$.

Lemma 6. *For* $\mathbf{v} \in \mathbb{R}^n$*, we have* $\mathbf{v} \in \mathcal{S}_{n,k}$ *if and only if* $\mathbf{G}_{\mathsf{partite}}\mathbf{v} = \mathbf{1}$ *and* $\mathbf{v}$ *is* k*-sparse.*

Proof. For the "only if" direction, observe that $\mathbf{v} \in \mathcal{S}_{n,k}$ directly implies that $\mathbf{v}$ is k-sparse and $\mathbf{G}_{\mathsf{partite}}\mathbf{v} = \mathbf{1}$, as each part of $\mathbf{v}$ sums to 1.

For the "if" direction, suppose $\mathbf{G}_{\mathsf{partite}}\mathbf{v} = \mathbf{1}$ and $\mathbf{v}$ is k-sparse. We need to show that $\mathbf{v}$ is both *partite* and *binary*. To see that $\mathbf{v}$ is partite, observe that the condition $\mathbf{G}_{\mathsf{partite}}\mathbf{v} = \mathbf{1}$ enforces the sum of the entrices of $\mathbf{v}$ in each of the k blocks to be 1. Therefore, each block has at least one non-zero entry, as otherwise that block would sum to 0. Since each of the k blocks has at least one non-zero entry, by sparsity, each block has exactly one non-zero entry, as otherwise, $\mathbf{v}$ would not be k-sparse. Since each block has exactly one non-zero entry that sums to 1, that entry must be 1. Therefore, $\mathbf{v} \in \mathcal{S}_{n,k}$.

Now, we prove the main theorem of this section, Theorem 4.

Proof (Proof of Theorem 4). Let $(\mathbf{B}, \mathbf{t} := \mathbf{Bz} + \mathbf{e})$ be our $\mathsf{BinaryBDD}_{d,\alpha}$ instance, where $\mathbf{B} \in \mathbb{R}^{d \times d}$, $\mathbf{z} \in \{-1, 1\}^d$ and $\|\mathbf{e}\|_2 \leq \alpha \cdot \lambda_{1,\mathsf{bin}}(\mathbf{B})$. Let $\mathbf{R} \in \mathbb{R}^{m_1 \times d}$ be a random matrix where each entry is drawn i.i.d. from $\mathcal{N}(0, 1)$. We assume we know a value $\widehat{\lambda}_1$ such that $\widehat{\lambda}_1 \in [\lambda_{1,\mathsf{bin}}(\mathbf{B}), 2\lambda_{1,\mathsf{bin}}(\mathbf{B}))$. (By Lemma 1, doubling search over $\widehat{\lambda}_1$ and taking the best BinaryBDD solution will take at most polynomial time.)

Our design matrix $\mathbf{X}$ and target vector $\mathbf{y}$ for k-SLR will be defined as follows:

$$\mathbf{X} = \begin{pmatrix} \mathbf{X}_1 \\ \mathbf{X}_2 \end{pmatrix} = \begin{pmatrix} \mathbf{RBG}_{\mathsf{sparse}} \\ \gamma \mathbf{G}_{\mathsf{partite}} \end{pmatrix} \in \mathbb{R}^{(m_1+k) \times n}, \qquad \mathbf{y} = \begin{pmatrix} \mathbf{Rt} \\ \gamma \mathbf{1} \end{pmatrix} \in \mathbb{R}^{m_1+k}.$$

Here, $\gamma \in \mathbb{R}$ is a scalar that will be set later. We define m_1 so that $m_1 + k = m$, which implies that $m_1 = m - k \geq m - d \geq 16d$. We define the scalar δ for the k-SLR instance so that

$$\delta^2 = \frac{3m_1 \cdot \widehat{\lambda}_1^2}{100m} = \Theta\left(\lambda_{1,\mathsf{bin}}(\mathbf{B})^2\right). \tag{1}$$

Invoking the k-SLR solver on the instance $(\mathbf{X}, \mathbf{y})$ with parameter δ, our reduction will get a solution $\widehat{\boldsymbol{\theta}}$ such that

$$\frac{\left\|\mathbf{X}\widehat{\boldsymbol{\theta}} - \mathbf{y}\right\|_2^2}{m} \leq \delta^2. \tag{2}$$

Next, we round $\widehat{\boldsymbol{\theta}}$ to the nearest integer entrywise to get $\mathsf{round}(\widehat{\boldsymbol{\theta}}) \in \mathbb{Z}^n$, and we output $\mathbf{G}_{\mathsf{sparse}}\mathsf{round}(\widehat{\boldsymbol{\theta}}) \in \mathbb{Z}^d$ as the $\mathsf{BinaryBDD}_{d,\alpha}$ solution.

First, we show completeness, in the sense that there exists k-sparse $\widehat{\boldsymbol{\theta}} \in \mathbb{R}^n$ such that (2) is satisfied. Recall from Lemma 5 that there exists a unique $\boldsymbol{\theta}^* \in \mathcal{S}_{n,k}$ such that $\mathbf{G}_{\mathsf{sparse}}\boldsymbol{\theta}^* = \mathbf{z}$. We will show that setting $\widehat{\boldsymbol{\theta}} = \boldsymbol{\theta}^*$ satisfies (2). We have

$$\mathbf{X}\boldsymbol{\theta}^* - \mathbf{y} = \begin{pmatrix} \mathbf{RBG}_{\mathsf{sparse}} \\ \gamma\mathbf{G}_{\mathsf{partite}} \end{pmatrix} \boldsymbol{\theta}^* - \begin{pmatrix} \mathbf{Rt} \\ \gamma\mathbf{1} \end{pmatrix} = \begin{pmatrix} \mathbf{RBG}_{\mathsf{sparse}}\boldsymbol{\theta}^* - \mathbf{Rt} \\ \gamma\mathbf{G}_{\mathsf{partite}}\boldsymbol{\theta}^* - \gamma\mathbf{1} \end{pmatrix} = \begin{pmatrix} \mathbf{RBz} - \mathbf{Rt} \\ \mathbf{0} \end{pmatrix},$$

where the last equality holds by Lemma 6 and definition of $\boldsymbol{\theta}^*$. Continuing the equality, we have

$$\mathbf{X}\boldsymbol{\theta}^* - \mathbf{y} = \begin{pmatrix} \mathbf{RBz} - \mathbf{R}(\mathbf{Bz} + \mathbf{e}) \\ \mathbf{0} \end{pmatrix} = \begin{pmatrix} -\mathbf{Re} \\ \mathbf{0} \end{pmatrix},$$

which implies $\left\|\mathbf{X}\widehat{\boldsymbol{\theta}} - \mathbf{y}\right\|_2^2 = \|\mathbf{Re}\|_2^2$. Since $\mathbf{R}$ was sampled independently of $\mathbf{e}$, observe that $\|\mathbf{Re}\|_2^2 \sim \|\mathbf{e}\|_2^2 \cdot \chi^2(m_1)$. By Lemma 3, it follows that

$$\Pr\left[\|\mathbf{Re}\|_2^2 \geq 5m_1/2 \cdot \|\mathbf{e}\|_2^2\right] \leq e^{-m_1/4}.$$

Therefore, with probability at least $1 - e^{-m_1/4}$, we have

$$\frac{\left\|\mathbf{X}\widehat{\boldsymbol{\theta}} - \mathbf{y}\right\|_2^2}{m} \leq \frac{5m_1 \|\mathbf{e}\|_2^2}{2m} \leq \frac{5m_1\alpha^2 \cdot \widehat{\lambda}_1^2}{2m} < \delta^2,$$

since $\alpha \leq 1/10$, as desired, showing that (2) is satisfied.

We now show soundness, which we will prove by contradiction. Suppose that $\mathbf{G}_{\mathsf{sparse}}\mathsf{round}(\widehat{\boldsymbol{\theta}}) = \mathbf{z}'$ for some $\mathbf{z}' \neq \mathbf{z}$ and (2) holds. Since (2) holds, in particular, we know $\left\|\gamma\mathbf{G}_{\mathsf{partite}}\widehat{\boldsymbol{\theta}} - \gamma\mathbf{1}\right\|_2^2 \leq m\delta^2$, or equivalently,

$$\left\|\mathbf{G}_{\mathsf{partite}}\widehat{\boldsymbol{\theta}} - \mathbf{1}\right\|_2 \leq \sqrt{m}\delta/\gamma.$$

Throughout, assume we set γ so that $\sqrt{m}\delta/\gamma < 1/2$. Each part of $\widehat{\boldsymbol{\theta}}$ must have exactly one non-zero entry, as $\widehat{\boldsymbol{\theta}}$ is k-sparse and cannot have any part with all zero entries. Moreover, we know this non-zero entry must be in $[1-\sqrt{m}\delta/\gamma, 1+\sqrt{m}\delta/\gamma]$, so by a choice of γ such that $\sqrt{m}\delta/\gamma < 1/2$, this entry must round to 1. Therefore, $\mathsf{round}(\widehat{\boldsymbol{\theta}}) \in \mathcal{S}_{n,k}$, and $\mathsf{round}(\widehat{\boldsymbol{\theta}})$ and $\widehat{\boldsymbol{\theta}}$ have the same support. This implies that $\mathbf{z}' = \mathbf{G}_{\mathsf{sparse}}\mathsf{round}(\widehat{\boldsymbol{\theta}}) \in \{-1,1\}^d$ and $\mathbf{G}_{\mathsf{partite}}\mathsf{round}(\widehat{\boldsymbol{\theta}}) = \mathbf{1}$ by Lemma 6. Moreover, since we can restrict to the support of $\widehat{\boldsymbol{\theta}}$, we have

$$\left\|\widehat{\boldsymbol{\theta}} - \mathsf{round}(\widehat{\boldsymbol{\theta}})\right\|_2 = \left\|\mathbf{G}_{\mathsf{partite}}\widehat{\boldsymbol{\theta}} - \mathbf{G}_{\mathsf{partite}}\mathsf{round}(\widehat{\boldsymbol{\theta}})\right\|_2 = \left\|\mathbf{G}_{\mathsf{partite}}\widehat{\boldsymbol{\theta}} - \mathbf{1}\right\|_2 \le \sqrt{m}\delta/\gamma.$$

Since $\widehat{\boldsymbol{\theta}} - \mathsf{round}(\widehat{\boldsymbol{\theta}})$ is also k-sparse and partite, it follows that

$$\left\|\mathbf{G}_{\mathsf{sparse}}\left(\widehat{\boldsymbol{\theta}} - \mathsf{round}(\widehat{\boldsymbol{\theta}})\right)\right\|_2 \le \frac{\delta\sqrt{dm}}{\gamma\sqrt{k}},$$

as each column of any block of $\mathbf{G}_{\mathsf{sparse}}$ has ℓ_2 norm exactly $\sqrt{d/k}$. Multiplying on the left by $\mathbf{B}$, we have

$$\left\|\mathbf{B}\mathbf{G}_{\mathsf{sparse}}\widehat{\boldsymbol{\theta}} - \mathbf{B}\mathbf{z}'\right\|_2 \le \sigma_{\max}(\mathbf{B}) \cdot \left\|\mathbf{G}_{\mathsf{sparse}}\left(\widehat{\boldsymbol{\theta}} - \mathsf{round}(\widehat{\boldsymbol{\theta}})\right)\right\|_2 \le \frac{\sigma_{\max}(\mathbf{B}) \cdot \delta\sqrt{dm}}{\gamma\sqrt{k}}. \tag{3}$$

On the other hand, by definition of $\lambda_{1,\mathsf{bin}}(\mathbf{B})$, since $\mathbf{z} \neq \mathbf{z}' \in \{-1,1\}^d$, we know

$$\|\mathbf{B}\mathbf{z} - \mathbf{B}\mathbf{z}'\|_2 \ge \lambda_{1,\mathsf{bin}}(\mathbf{B}). \tag{4}$$

By combining (3) and (4) by the triangle inequality, we have

$$\left\|\mathbf{B}\mathbf{G}_{\mathsf{sparse}}\widehat{\boldsymbol{\theta}} - \mathbf{B}\mathbf{z}\right\|_2 \ge \lambda_{1,\mathsf{bin}}(\mathbf{B}) - \frac{\sigma_{\max}(\mathbf{B}) \cdot \delta\sqrt{dm}}{\gamma\sqrt{k}}.$$

We now set

$$\gamma = \max\left(3\delta\sqrt{m}, \frac{100 \cdot \sigma_{\max}(\mathbf{B}) \cdot \delta\sqrt{dm}}{\sqrt{k} \cdot \widehat{\lambda}_1}\right) = \Theta\left(\frac{\sigma_{\max}(\mathbf{B}) \cdot \sqrt{dm}}{\sqrt{k}}\right), \tag{5}$$

so that

$$\left\|\mathbf{B}\mathbf{G}_{\mathsf{sparse}}\widehat{\boldsymbol{\theta}} - \mathbf{B}\mathbf{z}\right\|_2 \ge \lambda_{1,\mathsf{bin}}(\mathbf{B}) - \frac{\widehat{\lambda}_1}{100} \ge \frac{49}{50}\lambda_{1,\mathsf{bin}}(\mathbf{B}).$$

(We remark that γ is efficiently computable because $\sigma_{\max}(\mathbf{B})$ is efficiently computable and we assume $\widehat{\lambda}_1$ is known.) By incorporating the error $\mathbf{e}$ and multiplication on the left by $\mathbf{R}$, we have

$$\left\|\mathbf{X}\widehat{\boldsymbol{\theta}} - \mathbf{y}\right\|_2 \ge \left\|\mathbf{R}\mathbf{B}\mathbf{G}_{\mathsf{sparse}}\widehat{\boldsymbol{\theta}} - \mathbf{R}\left(\mathbf{B}\mathbf{z} + \mathbf{e}\right)\right\|_2 \ge \sigma_{\min}(\mathbf{R})\left(\frac{49}{50}\lambda_{1,\mathsf{bin}}(\mathbf{B}) - \|\mathbf{e}\|_2\right).$$

By combining this inequality with (2), we get a contradiction as long as

$$\delta\sqrt{m} \le \sigma_{\min}(\mathbf{R})\left(\frac{49}{50}\lambda_{1,\mathsf{bin}}(\mathbf{B}) - \|\mathbf{e}\|_2\right). \tag{6}$$

By Lemma 2, we know $\sigma_{\min}(\mathbf{R}) \geq \sqrt{m_1}/2$ with probability at least $1-2e^{-m_1/32}$, and since we also know $\|\mathbf{e}\|_2 \leq \alpha\lambda_{1,\mathsf{bin}}(\mathbf{B})$, we have

$$\begin{aligned}\sigma_{\min}(\mathbf{R})\left(\frac{49}{50}\lambda_{1,\mathsf{bin}}(\mathbf{B}) - \|\mathbf{e}\|_2\right) &\geq \frac{\sqrt{m_1}}{2}\left(\frac{49}{50}\lambda_{1,\mathsf{bin}}(\mathbf{B}) - \alpha\lambda_{1,\mathsf{bin}}(\mathbf{B})\right)\\ &= \lambda_{1,\mathsf{bin}}(\mathbf{B})\sqrt{m_1}\left(\frac{49}{100} - \frac{\alpha}{2}\right).\end{aligned}$$

On the other hand, we know

$$\delta\sqrt{m} = \frac{\widehat{\lambda}_1\sqrt{3m_1}}{10\sqrt{m}} \cdot \sqrt{m} \leq \frac{1}{5} \cdot \lambda_{1,\mathsf{bin}}(\mathbf{B})\sqrt{3m_1}.$$

Combining these inequalities, we get a contradiction if

$$\frac{\sqrt{3}}{5} \leq \frac{49}{100} - \frac{\alpha}{2},$$

which indeed holds if $\alpha \leq 1/10$.

4 Performance of Lasso on Our *k*-SLR Instances

4.1 Normalizing Our *k*-SLR Instances

First, we follow the convention of Negahban et al. [NRWY12] and normalize our design matrices according to Definition 4.

Lemma 7. *Let $\mathbf{X} \in \mathbb{R}^{m\times n}$ denote the design matrix we obtain in Theorem 4. Let $Z = \Theta(\sigma_{\max}(\mathbf{B}) \cdot \sqrt{d/k})$ be a real number. Then, with probability at least $1 - e^{-\Omega(m_1)}$, $\widetilde{\mathbf{X}} := \frac{1}{Z} \cdot \mathbf{X}$ is column-normalized, and so*

$$\max_{\substack{\|\mathbf{v}\|_2=1,\\ \|\mathbf{v}\|_0\leq 2k}} \|\mathbf{X}\mathbf{v}\|_2 \leq \Theta\left(\sigma_{\max}(\mathbf{B})\sqrt{dm}\right) \tag{7}$$

Proof. For our instance $\mathbf{X}$, we have

$$\|\mathsf{col}_i(\mathbf{X})\|_2^2 = \|\mathsf{col}_i(\mathbf{X}_1)\|_2^2 + \|\mathsf{col}_i(\mathbf{X}_2)\|_2^2 = \|\mathsf{col}_i(\mathbf{X}_1)\|_2^2 + \gamma^2.$$

Moreover,

$$\begin{aligned}\|\mathsf{col}_i(\mathbf{X}_1)\|_2 = \|\mathsf{col}_i(\mathbf{R}\mathbf{B}\mathbf{G}_{\mathsf{sparse}})\|_2 &= \|\mathbf{R}\mathbf{B}\mathsf{col}_i(\mathbf{G}_{\mathsf{sparse}})\|_2\\ &\leq \sigma_{\max}(\mathbf{R}) \cdot \sigma_{\max}(\mathbf{B}) \cdot \|\mathsf{col}_i(\mathbf{G}_{\mathsf{sparse}})\|_2\\ &= O\left(\sqrt{m_1} \cdot \sigma_{\max}(\mathbf{B}) \cdot \sqrt{d/k}\right),\end{aligned}$$

with probability at least $1 - e^{-\Omega(m_1)}$ over $\mathbf{R}$ by Lemma 2. Since our setting of γ (Eq. (5)) matches this up to constant factors, we have

$$\|\mathsf{col}_i(\mathbf{X})\|_2 = O\left(\sqrt{m} \cdot \sigma_{\max}(\mathbf{B}) \cdot \sqrt{d/k}\right).$$

Therefore, we can set a normalization factor $Z = \Theta(\sigma_{\max}(\mathbf{B}) \cdot \sqrt{d/k})$ such that

$$\widetilde{\mathbf{X}} := \frac{1}{Z} \cdot \mathbf{X}$$

is column-normalized. In particular, note that $\widetilde{\mathbf{X}}$ is scale invariant with respect to $\mathbf{B}$. By Lemma 4, we get that

$$\max_{\substack{\|\mathbf{v}\|_2=1,\\ \|\mathbf{v}\|_0 \le 2k}} \|\mathbf{X}\mathbf{v}\|_2 = Z \cdot \max_{\substack{\|\mathbf{v}\|_2=1,\\ \|\mathbf{v}\|_0 \le 2k}} \left\|\widetilde{\mathbf{X}}\mathbf{v}\right\|_2 \le Z \cdot \sqrt{2km} = \Theta\left(\sigma_{\max}(\mathbf{B})\sqrt{dm}\right)$$

4.2 Restricted Eigenvalue Condition

Now, we prove the lower bound on the restricted eigenvalues of the (column-normalized) design matrices of our k-SLR instances.

Lemma 8. *Suppose $d \ge 3k$. For a setting of $\epsilon = \Theta(k/d)$, $\widetilde{\mathbf{X}}$ satisfies the (ζ, ϵ)-restricted eigenvalue condition for all k-sparse, k-partite $S \subseteq [n]$ for some*

$$\zeta = \Theta\left(\frac{k}{d^4 \cdot \kappa(\mathbf{B})^2}\right),$$

with probability at least $1 - e^{-\Omega(m)}$.

Proof (Proof of Lemma 8). For simplicity, we will show that the unnormalized design matrix $\mathbf{X} = Z \cdot \widetilde{\mathbf{X}}$ has the property that for all k-sparse, k-partite $S \subset [n]$, and for all $\boldsymbol{\theta} \in \mathbb{C}_\epsilon(S)$,

$$\frac{\|\mathbf{X}\boldsymbol{\theta}\|_2^2}{m \cdot \|\boldsymbol{\theta}\|_2^2} \ge \zeta'.$$

We will then set $\zeta = \zeta'/Z^2$ to get the final bound.

Suppose for the sake of contradiction that there exists $\boldsymbol{\theta} \in \mathbb{C}_\epsilon(S)$ such that

$$\frac{\|\mathbf{X}\boldsymbol{\theta}\|_2^2}{m \cdot \|\boldsymbol{\theta}\|_2^2} < \zeta'.$$

Without loss of generality, we can scale $\boldsymbol{\theta}$ so that $\|\boldsymbol{\theta}\|_1 = 1$. In particular, observe that $\|\boldsymbol{\theta}\|_2^2 \le \|\boldsymbol{\theta}\|_1^2 = 1$. We have

$$\begin{aligned}\zeta' > \frac{\|\mathbf{X}\boldsymbol{\theta}\|_2^2}{m \cdot \|\boldsymbol{\theta}\|_2^2} &= \frac{\|\mathbf{R}\mathbf{B}\mathbf{G}_{\mathsf{sparse}}\boldsymbol{\theta}\|_2^2 + \|\mathbf{G}_{\mathsf{partite}}\boldsymbol{\theta}\|_2^2}{m \cdot \|\boldsymbol{\theta}\|_2^2} \\ &\ge \frac{1}{m}\left(\sigma_{\min}(\mathbf{R}\mathbf{B})^2\|\mathbf{G}_{\mathsf{sparse}}\boldsymbol{\theta}\|_2^2 + \|\mathbf{G}_{\mathsf{partite}}\boldsymbol{\theta}\|_2^2\right).\end{aligned}$$

If $\|\mathbf{G}_{\mathsf{partite}}\boldsymbol{\theta}\|_2^2 \ge \zeta' m$, we have arrived at a contradiction. Otherwise, we will show that $\|\mathbf{G}_{\mathsf{sparse}}\boldsymbol{\theta}\|_2^2 \ge \zeta' m/\sigma_{\min}(\mathbf{R}\mathbf{B})^2$, which would also be a contradiction.

Indeed, since $\mathbf{G}_{\mathsf{sparse}}\boldsymbol{\theta} \in \mathbb{R}^d$, we know $\|\mathbf{G}_{\mathsf{sparse}}\boldsymbol{\theta}\|_2 \sqrt{d} \geq \|\mathbf{G}_{\mathsf{sparse}}\boldsymbol{\theta}\|_1$, so it suffices to show that $\|\mathbf{G}_{\mathsf{sparse}}\boldsymbol{\theta}\|_1 \geq \tau_1$, where we define

$$\tau_1 := \frac{\sqrt{\zeta' dm}}{\sigma_{\min}(\mathbf{RB})}.$$

Since $\boldsymbol{\theta} \in \mathbb{C}_\epsilon(S)$ for some k-partite k-sparse $S \subseteq [n]$, we can write $\boldsymbol{\theta} = \boldsymbol{\theta}_S + \boldsymbol{\theta}_{\bar{S}}$ such that $\|\boldsymbol{\theta}_{\bar{S}}\|_1 \leq (1+\epsilon)\|\boldsymbol{\theta}_S\|_1$. Let $\eta^{(i)} \in \mathbb{R}$ be the ith nonzero entry of $\boldsymbol{\theta}_S$, and let $\bar{\boldsymbol{\eta}}^{(i)} \in \mathbb{R}^{n/k}$ be the ith part of $\boldsymbol{\theta}_{\bar{S}}$. Note that without loss of generality, we can assume that all the entries of $\boldsymbol{\theta}_S$ are non-negative, that is, for all i, $\eta^{(i)} \geq 0$. If not, suppose $\eta^{(i)} < 0$ for some $i \in [k]$. Because of the block diagonal structure of $\mathbf{G}_{\mathsf{sparse}}$, we can simply negate the ith part of $\boldsymbol{\theta}$ so that none of the norms change.

By applying Lemma 9 to the ℓ_1 norm of the parts of $\boldsymbol{\theta}_S$ and $\boldsymbol{\theta}_{\bar{S}}$, we know that there is some part $i \in [k]$ such that $\eta^{(i)} \geq \epsilon/k$ and $\|\bar{\boldsymbol{\eta}}^{(i)}\|_1 \leq (1+10\epsilon)\eta^{(i)}$. We will consider this part i, and we will write $\eta, \bar{\boldsymbol{\eta}}$ instead of $\eta^{(i)}, \bar{\boldsymbol{\eta}}^{(i)}$ for simplicity, i.e.,

$$\sum_j |\bar{\eta}_j| = \|\bar{\boldsymbol{\eta}}\|_1 \leq (1+10\epsilon)\eta. \tag{8}$$

We will provide a lower bound on $\|\mathbf{G}_{\mathsf{sparse}}\boldsymbol{\theta}\|_1$ considering only the ith part. Since $\|\mathbf{G}_{\mathsf{partite}}\boldsymbol{\theta}\|_2 \leq \sqrt{\zeta' m}$,

$$\eta + \sum_j \bar{\eta}_j \leq \left|\eta + \sum_j \bar{\eta}_j\right| \leq \frac{1}{\gamma}\|\mathbf{G}_{\mathsf{partite}}\boldsymbol{\theta}\|_2 \leq \frac{\sqrt{\zeta' m}}{\gamma}. \tag{9}$$

Adding (8) and (9), we get an upper bound on the sum of the positive entries in $\bar{\boldsymbol{\eta}}$:

$$\sum_j \bar{\eta}_j \cdot \mathbb{1}[\bar{\eta}_j > 0] \leq \frac{\sqrt{\zeta' m}}{2\gamma} + 5\epsilon\eta. \tag{10}$$

Let $\mathbf{H} \in \{-1,1\}^{d/k \times 2^{d/k}}$ be a block in $\mathbf{G}_{\mathsf{sparse}}$, i.e., whose columns are all of the vectors in $\{-1,1\}^{d/k}$. Let $\mathbf{h}^{(j)}$ denote the jth column of $\mathbf{H}$. Without loss of generality, we can choose the location of the η entry within the block so that it corresponds to the column $\mathbf{1} \in \mathbb{R}^{d/k}$. To see this, if not, we can flip the sign of the rows of $\mathbf{H}$ that correspond to -1 entries in the column corresponding to η. This has the effect of permuting the columns of $\mathbf{H}$ and has no effect on $\|\mathbf{G}_{\mathsf{sparse}}\boldsymbol{\theta}\|_1$ or $\|\mathbf{G}_{\mathsf{partite}}\boldsymbol{\theta}\|_2$ as the entries just flip sign. We will use the convention that first column of $\mathbf{H}$ is $\mathbf{1}$. By considering only part i, we have the inequality

$$\|\mathbf{G}_{\mathsf{sparse}}\boldsymbol{\theta}\|_1 \geq \left\|\eta\mathbf{1} + \sum_{j=2}^{2^{d/k}} \bar{\eta}_j \mathbf{h}^{(j)}\right\|_1 \geq \sum_{\ell=1}^{d/k}\left(\eta + \sum_{j=2}^{2^{d/k}} \bar{\eta}_j h_\ell^{(j)}\right) = \frac{\eta d}{k} + \sum_{j=2}^{2^{d/k}} \bar{\eta}_j \sum_{\ell=1}^{d/k} h_\ell^{(j)}. \tag{11}$$

Since $\mathbf{h}^{(j)} \neq \mathbf{1}$ for all $j \geq 2$, we know for all $j \geq 2$,

$$\sum_{\ell=1}^{d/k} h_\ell^{(j)} \leq \frac{d}{k} - 2,$$

as at least one entry in $\mathbf{h}^{(j)}$ is -1. This bound is helpful only when $\bar{\eta}_j < 0$, as this will let us lower bound $\|\mathbf{G}_{\mathsf{sparse}}\boldsymbol{\theta}\|_1$. Thankfully, (10) gives us a bound on the sum of the positive $\bar{\eta}_j$ contributions, in which case we use the trivial bound

$$\sum_{\ell=1}^{d/k} h_\ell^{(j)} \geq -\frac{d}{k}.$$

Let $J_+, J_- \subseteq [2^{d/k}] \setminus \{1\}$ denote the set of columns j where $\bar{\eta}_j \geq 0$ and $\bar{\eta}_j < 0$, respectively. Combining the above bounds with (11), we have

$$\|\mathbf{G}_{\mathsf{sparse}}\boldsymbol{\theta}\|_1 \geq \frac{\eta d}{k} + \sum_{j=2}^{2^{d/k}} \bar{\eta}_j \sum_{\ell=1}^{d/k} = \frac{\eta d}{k} + \sum_{j \in J_+} \bar{\eta}_j \sum_{\ell=1}^{d/k} h_\ell^{(j)} + \sum_{j \in J_-} \bar{\eta}_j \sum_{\ell=1}^{d/k} h_\ell^{(j)} \tag{12}$$

$$\geq \frac{\eta d}{k} - \frac{d}{k} \sum_{j \in J_+} \bar{\eta}_j - \left(\frac{d}{k} - 2\right) \left| \sum_{j \in J_-} \bar{\eta}_j \right| \tag{13}$$

$$\geq \frac{\eta d}{k} - \frac{d}{k} \left(\frac{\sqrt{\zeta' m}}{2\gamma} + 5\epsilon\eta \right) - \left(\frac{d}{k} - 2\right) \sum_{j=2}^{2^{d/k}} |\bar{\eta}_j| \tag{14}$$

$$\geq \frac{\eta d}{k} - \frac{d}{k} \left(\frac{\sqrt{\zeta' m}}{2\gamma} + 5\epsilon\eta \right) - \left(\frac{d}{k} - 2\right) (1 + 10\epsilon)\eta, \tag{15}$$

where the penultimate inequality comes from (10), and the last inequality comes from (8). We now bound the remaining terms. Setting

$$\epsilon := \frac{k}{50d},$$

and assuming

$$\gamma \geq \frac{5d\sqrt{\zeta' m}}{\eta k}, \tag{16}$$

we have the guarantee that

$$\frac{\sqrt{\zeta' m}}{2\gamma} + 5\epsilon\eta \leq \frac{\eta k}{10d} + \frac{\eta k}{10d} = \frac{\eta k}{5d}.$$

Furthermore, our setting of ϵ guarantees

$$\left(\frac{d}{k} - 2\right)(1 + 10\epsilon) = \frac{d}{k} - \frac{9}{5} - \frac{2k}{5d} < \frac{d}{k} - 1.$$

Plugging these into (15), we have

$$\begin{aligned}\|\mathbf{G}_{\mathsf{sparse}}\boldsymbol{\theta}\|_1 &\geq \frac{\eta d}{k} - \frac{d}{k}\left(\frac{\sqrt{\zeta' m}}{2\gamma} + 5\epsilon\eta\right) - \left(\frac{d}{k} - 2\right)(1+10\epsilon)\eta \\ &\geq \frac{\eta d}{k} - \frac{d}{k}\cdot\frac{\eta k}{5d} - \left(\frac{d}{k} - 1\right)\eta \\ &= \frac{4}{5}\eta \geq \frac{4\epsilon}{5k} = \frac{2}{125d},\end{aligned}$$

where the last inequality comes from our application of Lemma 9, which implies $\eta \geq \epsilon/k$. Recall that we get a contradiction if

$$\|\mathbf{G}_{\mathsf{sparse}}\boldsymbol{\theta}\|_1 \geq \tau_1 := \frac{\sqrt{\zeta' dm}}{\sigma_{\min}(\mathbf{RB})}.$$

Therefore, setting ζ' so that

$$\frac{\sqrt{\zeta' dm}}{\sigma_{\min}(\mathbf{RB})} < \frac{2}{125d}$$

arrives us at a contradiction. Since $\sigma_{\min}(\mathbf{RB}) \geq \sigma_{\min}(\mathbf{R})\sigma_{\min}(\mathbf{B}) \geq \sqrt{m_1}/2 \cdot \sigma_{\min}(\mathbf{B})$ with probability at least $1 - e^{-\Omega(m_1)}$ by Lemma 2, it suffices to set

$$\zeta' := \frac{1}{2}\cdot\frac{m_1\sigma_{\min}(\mathbf{B})^2}{125^2 d^3\, m} = \Theta\left(\frac{\sigma_{\min}(\mathbf{B})^2}{d^3}\right).$$

Now, we justify our assumption on γ in (16). Recall that our reduction sets

$$\gamma = \Theta\left(\frac{\sigma_{\max}(\mathbf{B})\sqrt{dm}}{\sqrt{k}}\right).$$

By our setting of ζ' and since $\eta \geq \epsilon/k = 1/(50d)$, we have

$$\gamma \gg \frac{\sigma_{\min}(\mathbf{B})\sqrt{dm}}{k} \geq \Theta\left(\frac{\sigma_{\min}(\mathbf{B})\sqrt{m}}{k\eta\sqrt{d}}\right) = \Theta\left(\frac{d\sqrt{\zeta' m}}{\eta k}\right),$$

showing that (16) is indeed satisfied.

Finally, scaling by Z^2, we have

$$\zeta = \frac{\zeta'}{Z^2} = \Theta\left(\frac{k\cdot\sigma_{\min}(\mathbf{B})^2}{d^4\cdot\sigma_{\max}(\mathbf{B})^2}\right),$$

as desired.

Lemma 9. *Suppose $x_i, y_i \in \mathbb{R}_{\geq 0}$ for $i \in [k]$, such that $\sum_{i=1}^k x_i + \sum_{i=1}^k y_i = 1$. Further suppose that for some $\epsilon \in (0, 1/20)$, $\sum_{i=1}^k y_i \leq (1+\epsilon)\sum_{i=1}^k x_i$. Then, there is some $i^* \in [k]$ such that $x_{i^*} \geq \epsilon/k$ and $y_{i^*} \leq (1+10\epsilon)x_{i^*}$.*

Proof. Since $\sum_{i=1}^{k} x_i + \sum_{i=1}^{k} y_i = 1$ and $\sum_{i=1}^{k} y_i \leq (1+\epsilon)\sum_{i=1}^{k} x_i$, this implies that $\sum_{i=1}^{k} x_i \geq \frac{1}{2+\epsilon} \geq \frac{1}{2} - \epsilon$, and $\sum_{i=1}^{k} y_i \leq \frac{1}{2} + \epsilon$.

Suppose for contradiction that for all $i \in [k]$, either $x_i < \epsilon/k$ or $y_i > (1 + 10\epsilon)x_i$. Let A be the set of all $i \in [k]$ such that $x_i < \epsilon/k$, and let $B = [k] \setminus A$. By assumption, for all $i \in B$, we have that $y_i > (1+10\epsilon)x_i$. Then, since $\sum_{i\in A} x_i \leq \epsilon$,

$$\begin{aligned}\sum_{i\in B} y_i > (1+10\epsilon)\sum_{i\in B} x_i &= (1+10\epsilon)\left(\sum_{i\in[k]} x_i - \sum_{i\in A} x_i\right) \\ &\geq (1+10\epsilon)\left(\frac{1}{2} - 2\epsilon\right) = \frac{1}{2} + 5\epsilon - 2\epsilon - 20\epsilon^2 \geq \frac{1}{2} + 2\epsilon.\end{aligned}$$

This is a contradiction, since we also know that $\sum_{i\in B} y_i \leq \sum_{i\in[k]} y_i \leq \frac{1}{2} + \epsilon$.

4.3 Prediction Error Achieved by Lasso on Our Instances

We now apply Theorem 3 to our instance as generated in Theorem 4, to obtain a bound on the prediction error achieved by Lasso on our instances. Note that this is a mean squared prediction error bound as opposed to a mean squared error bound. To translate Lasso into an algorithm for $\mathsf{BinaryBDD}_{d,\alpha}$ via the reduction in Theorem 4, we can bound the mean-squared error by the triangle inequality (see Corollary 1).

Lemma 10. *Consider the instances* $(\mathbf{X}, \mathbf{y}, \delta = \Theta(\lambda_{1,\mathsf{bin}}(\mathbf{B}))$ *of* k-SLR *instances obtained in the reduction from a* $\mathsf{BinaryBDD}_{d,\alpha}$ *instance* $(\mathbf{B}, \cdot)$ *as in Theorem 4. On these instances, the thresholded Lasso estimator* $\boldsymbol{\theta}_{\mathsf{TL}}$ *can achieve prediction error bounded as*

$$\frac{1}{m}\left\|\mathbf{X}\widehat{\boldsymbol{\theta}}_{\mathsf{TL}} - \mathbf{X}\boldsymbol{\theta}^*\right\|_2^2 \leq O\left(\frac{\alpha^2 \cdot \lambda_{1,\mathsf{bin}}(\mathbf{B})^2 \cdot \kappa(\mathbf{B})^4 \cdot d^{10}}{k^2}\right).$$

Proof. With the goal of applying Theorem 3, we first bound $\|\widetilde{\mathbf{w}}\|_2$. On our instances, we have

$$\widetilde{\mathbf{w}} = \widetilde{\mathbf{y}} - \widetilde{\mathbf{X}}\boldsymbol{\theta}^* = \frac{1}{Z}\begin{pmatrix}\mathbf{Re} \\ \mathbf{0}\end{pmatrix}.$$

Therefore,

$$\|\widetilde{\mathbf{w}}\|_2 = \frac{1}{Z}\|\mathbf{Re}\|_2 \leq \frac{1}{Z}\sigma_{\max}(\mathbf{R})\|\mathbf{e}\|_2 \leq O\left(\frac{\alpha \cdot \lambda_{1,\mathsf{bin}}(\mathbf{B})\sqrt{m}}{Z}\right)$$

with probability at least $1 - e^{-\Omega(m)}$ over $\mathbf{R}$ by Lemma 2. Since $Z = \Theta(\sigma_{\max}(\mathbf{B}) \cdot \sqrt{d/k})$, we have

$$\|\widetilde{\mathbf{w}}\|_2 \leq O\left(\frac{\alpha \cdot \lambda_{1,\mathsf{bin}}(\mathbf{B})\sqrt{mk}}{\sigma_{\max}(\mathbf{B})\sqrt{d}}\right).$$

With this bound on $\|\widetilde{\mathbf{w}}\|_2$, we can plug in Lemma 8, which says that for $\epsilon = \Theta(k/d)$, $\widetilde{\mathbf{X}}$ satisfies the (ζ, ϵ)-restricted eigenvalue condition for all k-sparse, k-partite $S \subseteq [n]$ for

$$\zeta = \Theta\left(\frac{k}{d^4 \cdot \kappa(\mathbf{B})^2}\right).$$

Plugging in $\|\widetilde{\mathbf{w}}\|_2$, ϵ, and ζ into Theorem 3, we get

$$\frac{1}{m}\|\widetilde{\mathbf{X}}\widehat{\boldsymbol{\theta}}_{\mathsf{TL}} - \widetilde{\mathbf{X}}\boldsymbol{\theta}^*\|_2^2 \leq O\left(\frac{\|\widetilde{\mathbf{w}}\|_2^2 \cdot k^2}{\zeta^2 \cdot \epsilon^2 \cdot m}\right) \leq O\left(\frac{\alpha^2 \cdot \lambda_{1,\mathsf{bin}}(\mathbf{B})^2 \cdot \kappa(\mathbf{B})^4 \cdot d^9}{k \cdot \sigma_{\max}(\mathbf{B})^2}\right).$$

Scaling up by $Z^2 = \Theta(\sigma_{\max}(\mathbf{B})^2 \cdot d/k)$ to convert back to the unnormalized version, we have

$$\frac{1}{m}\|\mathbf{X}\widehat{\boldsymbol{\theta}}_{\mathsf{TL}} - \mathbf{X}\boldsymbol{\theta}^*\|_2^2 \leq O\left(\frac{\alpha^2 \cdot \lambda_{1,\mathsf{bin}}(\mathbf{B})^2 \cdot \kappa(\mathbf{B})^4 \cdot d^{10}}{k^2}\right),$$

as desired.

Corollary 1 (Lasso-based Algorithm for $\mathsf{BinaryBDD}_{d,\alpha}$). *Let d, k, m be integers such that k divides d and $m \geq \Omega(d)$. Let α be a real number such that*

$$\alpha \leq C \cdot \frac{k}{d^5 \cdot \kappa(\mathbf{B})^2}$$

for some universal constant $C > 0$. Then, running the reduction in Theorem 4 and reducing to Lasso as the k-SLR solver gives a $\mathrm{poly}(m, k \cdot 2^{d/k})$-time reduction from $\mathsf{BinaryBDD}_{d,\alpha}$ that succeeds with probability at least $1 - e^{-\Omega(m)}$. Setting k and m such that $d = 10k$ and $m = 100d$, this becomes a $\mathrm{poly}(d)$ time reduction from $\mathsf{BinaryBDD}_{d,\alpha}$ to Lasso that succeeds with probability at least $1 - e^{-\Omega(d)}$ as long as

$$\alpha \leq C' \cdot \frac{1}{d^4 \cdot \kappa(\mathbf{B})^2},$$

for some universal constant $C' > 0$.

Proof. Let $(\mathbf{X}, \mathbf{y} = \mathbf{X}\boldsymbol{\theta}^* + \mathbf{w}, \delta)$ be the k-SLR instance obtained in the reduction in Theorem 4. Then, if $\widehat{\boldsymbol{\theta}}_{\mathsf{TL}}$ is the solution obtained via thresholded Lasso, since $\|\mathbf{X}\boldsymbol{\theta}^* - \mathbf{y}\|_2^2/m = \|\mathbf{Re}\|_2^2/m \leq O(\alpha^2 \cdot \lambda_{1,\mathsf{bin}}(\mathbf{B}))$ with probability at least $1 - e^{-\Omega(m)}$, by triangle inequality, we have that

$$\begin{aligned}\frac{1}{m}\|\mathbf{X}\widehat{\boldsymbol{\theta}}_{\mathsf{TL}} - \mathbf{y}\|_2^2 &\leq O\left(\frac{1}{m}\|\mathbf{X}\widehat{\boldsymbol{\theta}}_{\mathsf{TL}} - \mathbf{X}\boldsymbol{\theta}^*\|_2^2 + \frac{1}{m}\|\mathbf{X}\boldsymbol{\theta}^* - \mathbf{y}\|_2^2\right) \\ &\leq O\left(\alpha^2 \cdot \lambda_{1,\mathsf{bin}}(\mathbf{B})^2 \cdot \kappa(\mathbf{B})^4 \cdot \frac{d^{10}}{k^2}\right) + O(\alpha^2 \cdot \lambda_{1,\mathsf{bin}}(\mathbf{B})^2).\end{aligned}$$

Theorem 4 needs a k-SLR solver with $\delta = \Theta(\lambda_{1,\mathsf{bin}}(\mathbf{B}))$ by (1), so setting

$$\alpha \leq C \cdot \frac{k}{d^5 \cdot \kappa(\mathbf{B})^2}$$

for some universal constant $C > 0$ suffices for the correctness of the reduction. This gives us a Lasso-based algorithm for $\mathsf{BinaryBDD}_{d,\alpha}$ that runs in time $\mathrm{poly}(m, k \cdot 2^{d/k})$.

Remark 3. We briefly remark that the information-theoretic k-SLR solver (i.e., by the bound in Proposition 1) would solve our BDD instance for sufficiently small *constant* α. In particular, if allowed oracle calls to an information-theoretic k-SLR solver, there would be a $\mathrm{poly}(d)$-time algorithm for $\mathsf{BinaryBDD}_{d,\alpha}$ for sufficiently small constant α.

Finally, we state how a hypothetical improvement to Lasso gives a corresponding parameter improvement to $\mathsf{BinaryBDD}_{d,\alpha}$.

Theorem 5. *Let d, k, m, n be integers such that k divides d, $m \geq \Omega(d)$ and $n = k \cdot 2^{d/k}$. Suppose there is a polynomial-time algorithm that takes k-SLR instances $(\widetilde{\mathbf{X}}, \widetilde{\mathbf{y}} = \widetilde{\mathbf{X}}\boldsymbol{\theta}^* + \widetilde{\mathbf{w}})$ with design matrices $\widetilde{\mathbf{X}} \in \mathbb{R}^{m \times n}$ that satisfy the (ζ, ϵ)-restricted eigenvalue condition for* $\mathrm{support}(\boldsymbol{\theta}^*)$*, and outputs a k-sparse vector $\widehat{\boldsymbol{\theta}}$ such that the prediction error is bounded as*

$$\frac{1}{m}\|\widetilde{\mathbf{X}}\widehat{\boldsymbol{\theta}} - \widetilde{\mathbf{X}}\boldsymbol{\theta}^*\|_2^2 \leq O\left(\frac{\|\widetilde{\mathbf{w}}\|_2^2 \cdot k^2}{\zeta^{2-\beta} \cdot \epsilon^2 \cdot m}\right)$$

for some constant $\beta \in (0, 2]$. Then there is an algorithm that runs in time $\mathrm{poly}(m, k \cdot 2^{d/k})$ that solves $\mathsf{BinaryBDD}_{d,\alpha}$ with probability $1 - e^{-\Omega(m)}$ on instances $(\mathbf{B}, \cdot)$ as long as

$$\alpha \leq C \cdot \frac{k^{1-\beta/2}}{d^{5-2\beta} \cdot \kappa(\mathbf{B})^{2-\beta}},$$

for some universal constant $C > 0$.

Proof (Proof of Theorem 5). The proof follows directly by modifying the exponent on ζ in Lemma 10 and Corollary 1.

5 Reduction from CLWE

We now define the continuous learning with errors (CLWE) assumption as defined by [BRST21], which holds assuming the quantum [BRST21] or classical [GVV22] hardness of worst-case lattice problems for various parameter regimes. We define $\mathbb{S}^{m-1}$ to be the unit sphere in $\mathbb{R}^m$, i.e., the set of all unit vectors in $\mathbb{R}^m$ according to the ℓ_2 norm. We write $\mathbf{v} \sim \mathbb{S}^{m-1}$ to denote sampling $\mathbf{v}$ from the uniform distribution over the sphere. Furthermore, for a vector $\mathbf{v} \in \mathbb{R}^n$, by $\mathbf{v} \pmod 1$, we mean the representative $\widetilde{\mathbf{v}} \in [-1/2, 1/2)^n$ such that $\mathbf{v} - \widetilde{\mathbf{v}} \in \mathbb{Z}^n$. Moreover, by this choice of representative, note that we have $|a \pmod 1| \leq |a|$ for all $a \in \mathbb{R}$.

Definition 7 (CLWE Assumption [BRST21]). *For a secret vector* $\mathbf{s} \sim \gamma_{\mathsf{CLWE}} \cdot \mathbb{S}^{m-1}$ *and error vector* $\mathbf{e} \sim \mathcal{N}(0, \beta^2 I_{n\times n})$*, we have the computational indistinguishability*

$$\begin{aligned}&\left(\mathbf{A} \sim \mathcal{N}(0,1)^{m\times n}, \mathbf{b}^\top := \mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top \pmod 1\right)\\ \approx &\left(\mathbf{A} \sim \mathcal{N}(0,1)^{m\times n}, \mathbf{b}^\top \sim [-1/2, 1/2)^m\right),\end{aligned}$$

up to constant advantage for algorithms running in time T*.*

This assumption is parameterized by $m, n, \gamma_{\mathsf{CLWE}}, \beta$ and T. For example, for the parameter setting $\gamma_{\mathsf{CLWE}} = 2\sqrt{m}$ and $\beta = 1/\operatorname{poly}(m)$, there are no algorithms known that run in time $T = 2^{o(m)}$ with $n = 2^{o(m)}$ samples.

Theorem 6. *Suppose* $k = m^\epsilon$ *for some* $\epsilon \in (0,1)$*. There is a* $\operatorname{poly}(n,m)$*-time reduction from* CLWE *in dimension* m *with* n *samples and parameters* $\gamma_{\mathsf{CLWE}} = 2\sqrt{m}, \beta = o(1/\sqrt{k})$ *to* k-SLR*, for* $n = k \cdot 2^{\Theta(m/k \cdot \log m)}$*. If the* k-SLR *solver runs in time* T*, the distinguisher for* CLWE *runs in time* $T + \operatorname{poly}(n,m)$*. Moreover, the reduction generates design matrices* $\mathbf{X}$ *that can be decomposed as*

$$\mathbf{X} = \begin{pmatrix}\mathbf{X}_1\\ \mathbf{X}_2\end{pmatrix},$$

where the distribution of each row of $\mathbf{X}_1 \in \mathbb{R}^{m\times n}$ *is* i.i.d. $\mathcal{N}(\mathbf{0}, I_{n\times n})$*, and* $\mathbf{X}_2 \in \mathbb{R}^{k\times n}$ *is proportional to a fixed, instance-independent matrix that depends only on* n *and* k*.*

The interpretations of Sect. 3.1 also apply here as to how one could remove the worst-case sub-matrix $\mathbf{X}_2$.

Proof. We first start with the CLWE assumption (see Definition 7), which says that for secret vector $\mathbf{s} \sim \gamma_{\mathsf{CLWE}} \cdot \mathbb{S}^{m-1}$ and error vector $\mathbf{e} \sim \mathcal{N}(0, \beta^2 I_{n\times n})$, we have the indistinguishability

$$\begin{aligned}&\left(\mathbf{A} \sim \mathcal{N}(0,1)^{m\times n}, \mathbf{b}^\top := \mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top \pmod 1\right)\\ \approx &\left(\mathbf{A} \sim \mathcal{N}(0,1)^{m\times n}, \mathbf{b}^\top \sim [-1/2, 1/2)^m\right).\end{aligned}$$

We construct our design matrix $\mathbf{X} \in \mathbb{R}^{(m+k)\times n}$ and target $\mathbf{y} \in \mathbb{R}^{m+k}$ as

$$\mathbf{X} := \begin{pmatrix}\mathbf{A}\\ \alpha \mathbf{G}_{\mathsf{partite}}\end{pmatrix}, \quad \mathbf{y} = \begin{pmatrix}\mathbf{0}\\ \alpha \mathbf{1}\end{pmatrix},$$

with parameter

$$\delta := \frac{1}{100\gamma_{\mathsf{CLWE}}\sqrt{m+k}} = \Theta\left(\frac{1}{\gamma_{\mathsf{CLWE}}\sqrt{m+k}}\right).$$

For a (k-sparse) k-SLR solution $\widehat{\boldsymbol{\theta}} \in \mathbb{R}^n$ to the instance $(\mathbf{X}, \mathbf{y}, \delta)$, we compute $\mathsf{round}(\widehat{\boldsymbol{\theta}}) \in \mathbb{Z}^n$ and check if

$$\left|\mathbf{b}^\top \mathsf{round}(\widehat{\boldsymbol{\theta}}) \pmod 1\right| < \frac{1}{4}.$$

If so, we output 1 (indicating CLWE), and otherwise, we output 0 (indicating null).

Soundness. To see that we have a distinguisher for CLWE, suppose $\widehat{\boldsymbol{\theta}} \in \mathbb{R}^n$ is a k-SLR solution, i.e.,

$$\frac{\left\|\mathbf{X}\widehat{\boldsymbol{\theta}} - \mathbf{y}\right\|_2^2}{m+k} \le \delta^2. \tag{17}$$

In particular, we know $\left\|\alpha \mathbf{G}_{\mathsf{partite}}\widehat{\boldsymbol{\theta}} - \alpha \mathbf{1}\right\|_2^2 \le (m+k)\delta^2 = \frac{1}{100^2\gamma_{\mathsf{CLWE}}^2}$, or equivalently,

$$\left\|\mathbf{G}_{\mathsf{partite}}\widehat{\boldsymbol{\theta}} - \mathbf{1}\right\|_2 \le \frac{1}{100\alpha\gamma_{\mathsf{CLWE}}}.$$

Throughout, assume we set α so that $100\alpha\gamma_{\mathsf{CLWE}} > 2$. Each part of $\widehat{\boldsymbol{\theta}}$ must have exactly one non-zero entry, as $\widehat{\boldsymbol{\theta}}$ is k-sparse and cannot have any part with all zero entries. Moreover, we know this non-zero entry must be in $[1 - 1/(100\alpha\gamma_{\mathsf{CLWE}}), 1 + 1/(100\alpha\gamma_{\mathsf{CLWE}})]$, so this entry must round to 1. Therefore, $\mathsf{round}(\widehat{\boldsymbol{\theta}}) \in \mathcal{S}_{n,k}$, and $\mathsf{round}(\widehat{\boldsymbol{\theta}})$ and $\widehat{\boldsymbol{\theta}}$ have the same support. By considering the support of $\widehat{\boldsymbol{\theta}}$, this implies

$$\left\|\mathsf{round}(\widehat{\boldsymbol{\theta}}) - \widehat{\boldsymbol{\theta}}\right\|_2 = \left\|\mathbf{G}_{\mathsf{partite}}\widehat{\boldsymbol{\theta}} - \mathbf{1}\right\|_2 \le \frac{1}{100\alpha\gamma_{\mathsf{CLWE}}}.$$

Moreover, we will set $\alpha = \max(\sqrt{n}, 3/(100\gamma_{\mathsf{CLWE}}))$ so that

$$\left\|\mathsf{round}(\widehat{\boldsymbol{\theta}}) - \widehat{\boldsymbol{\theta}}\right\|_2 < \frac{1}{100\gamma_{\mathsf{CLWE}}\sqrt{n}}.$$

By (17), we also know

$$\left\|\mathbf{A}\widehat{\boldsymbol{\theta}}\right\|_2 \le \delta\sqrt{m+k} \le \frac{1}{100\gamma_{\mathsf{CLWE}}}.$$

First, suppose we are the non-null case, i.e., $\mathbf{b}^\top = \mathbf{s}^\top\mathbf{A} + \mathbf{e}^\top$. Since $\mathsf{round}(\widehat{\boldsymbol{\theta}}) \in \mathbb{Z}^n$, we then have

$$\begin{aligned}
\left|\mathbf{b}^\top \mathsf{round}(\widehat{\boldsymbol{\theta}}) \pmod 1\right| &= \left|\mathbf{s}^\top \mathbf{A}\mathsf{round}(\widehat{\boldsymbol{\theta}}) + \mathbf{e}^\top \mathsf{round}(\widehat{\boldsymbol{\theta}}) \pmod 1\right| \\
&\le \left|\mathbf{s}^\top \mathbf{A}(\mathsf{round}(\widehat{\boldsymbol{\theta}}) - \widehat{\boldsymbol{\theta}})\right| + \left|\mathbf{s}^\top \mathbf{A}\widehat{\boldsymbol{\theta}}\right| + \left|\mathbf{e}^\top \mathsf{round}(\widehat{\boldsymbol{\theta}})\right| \\
&\le \left\|\mathbf{s}^\top \mathbf{A}\right\|_2 \cdot \left\|\mathsf{round}(\widehat{\boldsymbol{\theta}}) - \widehat{\boldsymbol{\theta}}\right\|_2 + \frac{1}{100} + \left|\mathbf{e}^\top \mathsf{round}(\widehat{\boldsymbol{\theta}})\right|.
\end{aligned}$$

Since $\mathbf{X}$ and $\mathbf{y}$ are independent of $\mathbf{e}$, and since $\mathsf{round}(\widehat{\boldsymbol{\theta}}) \in \mathcal{S}_{n,k}$, we know $\mathbf{e}^\top \mathsf{round}(\widehat{\boldsymbol{\theta}}) \sim \mathcal{N}(0, k\beta^2)$. By standard tails bounds on the Gaussian, we have $|\mathbf{e}^\top \mathsf{round}(\widehat{\boldsymbol{\theta}})| \le O(\beta\sqrt{k})$ with probability at least $99/100$. Similarly, since $\mathbf{s}$ and $\mathbf{A}$ are independent, we know $\mathbf{s}^\top \mathbf{A} \sim \mathcal{N}(0, \gamma_{\mathsf{CLWE}} I_{n\times n})$, so by Lemma 3, we know $\left\|\mathbf{s}^\top \mathbf{A}\right\|_2 \le 10\gamma_{\mathsf{CLWE}}\sqrt{n}$ with probability at least $1 - e^{-\Omega(n)}$. Plugging these into our previous bound, for $\beta = o(1/\sqrt{k})$,

$$\begin{aligned}\left|\mathbf{b}^\top \mathsf{round}(\widehat{\boldsymbol{\theta}}) \pmod 1\right| &\le \left\|\mathbf{s}^\top \mathbf{A}\right\|_2 \cdot \left\|\mathsf{round}(\widehat{\boldsymbol{\theta}}) - \widehat{\boldsymbol{\theta}}\right\|_2 + \frac{1}{100} + \left|\mathbf{e}^\top \mathsf{round}(\widehat{\boldsymbol{\theta}})\right| \\ &\le \frac{1}{10} + \frac{1}{100} + o(1) < \frac{1}{4},\end{aligned}$$

with probability at least $\frac{49}{50}$.

On the other hand, if we are in the null case, then $\mathbf{b} \sim [-1/2, 1/2)^m$ independently of $\widehat{\boldsymbol{\theta}}$, and consequently $\mathbf{b}^\top \mathsf{round}(\widehat{\boldsymbol{\theta}}) \pmod 1 \sim [-1/2, 1/2)$, in which case $\Pr\left[\left|\mathbf{b}^\top \mathsf{round}(\widehat{\boldsymbol{\theta}}) \pmod 1\right| \ge 1/4\right] = 1/2$.

Therefore, we have a distinguisher for CLWE with $\Omega(1)$ advantage.

Completeness. As a warm-up, we compute the expected number of solutions $\boldsymbol{\theta} \in \mathcal{S}_{n,k}$. In particular, since $\boldsymbol{\theta} \in \mathcal{S}_{n,k}$, we have $\mathbf{G}_{\mathsf{partite}}\boldsymbol{\theta} = \mathbf{1}$ by Lemma 6. It therefore suffices to show

$$\frac{\|\mathbf{A}\boldsymbol{\theta}\|_2^2}{m+k} \le \delta^2$$

for $\boldsymbol{\theta}$ to be a solution. For simplicity, let $\delta' = \delta\sqrt{m+k} = \frac{1}{100\gamma_{\mathsf{CLWE}}} = \frac{1}{200\sqrt{m}}$. Fix some $\boldsymbol{\theta} \in \mathcal{S}_{n,k}$. The random variable $\mathbf{A}\boldsymbol{\theta} \in \mathbb{R}^m$ (over the randomness of $\mathbf{A}$) is distributed according to $\mathcal{N}(\mathbf{0}, kI_{m\times m})$. Therefore, $\|\mathbf{A}\boldsymbol{\theta}\|_2^2 / k \sim \chi^2(m)$. As a result,

$$p := \Pr_{\mathbf{A}}\left[\|\mathbf{A}\boldsymbol{\theta}\|_2^2 \le (\delta')^2\right] = \Pr_{Z\sim\chi^2(m)}\left[Z \le \frac{(\delta')^2}{k}\right].$$

Using the CDF of the $\chi^2(m)$ distribution and the lower incomplete gamma function $\gamma(\cdot,\cdot)$, we have

$$\begin{aligned}p = \Pr_{Z\sim\chi^2(m)}\left[Z \le \frac{(\delta')^2}{k}\right] &= \frac{1}{\Gamma(m/2)} \cdot \gamma\left(\frac{m}{2}, \frac{(\delta')^2}{2k}\right) \\ &= \frac{1}{(m/2-1)!}\left(\frac{(\delta')^2}{2k}\right)^{m/2} e^{-(\delta')^2/2k} \cdot \\ &\qquad \sum_{j=0}^{\infty} \frac{\left(\frac{(\delta')^2}{2k}\right)^j}{(m/2)(m/2+1)\cdots(m/2+j)} \\ &= (1 \pm o(1)) \cdot \frac{(\delta')^m}{(m/2-1)!(2k)^{m/2}} \cdot \frac{2}{m}, \\ &= \left(\frac{\delta'}{\sqrt{mk}}\right)^{m(1\pm o(1))}\end{aligned}$$

because in our setting, $(\delta')^2/2k = o(1)$, so $e^{-(\delta')^2/2k} = 1 - o(1)$ and the infinite sum is dominated by its first term, and because $\delta' = o(1)$ and $\sqrt{mk} = \omega(1)$. Therefore, by linearity of expectation, the expected number of solutions X in $\mathcal{S}_{n,k}$ satisfies

$$\mathbb{E}[X] = \left(\frac{n}{k}\right)^k \cdot p.$$

Setting n so that

$$p = \left(\frac{n}{k}\right)^{-k/10}$$

gives

$$\mathbb{E}[X] = \left(\frac{n}{k}\right)^{9k/10}, \quad n = k \cdot 2^{(10 \pm o(1))m/k \cdot \log(\sqrt{mk}/\delta')} = k \cdot 2^{\Theta(m/k \cdot \log(\sqrt{mk} \cdot \gamma_{\mathsf{CLWE}}))}.$$

We turn this expected value bound into a $1 - o(1)$ bound on the existence of a sparse solution in the full version [GVV24].[8]

Acknowledgments. We greatly thank Kiril Bangachev for the proof of completeness in Theorem 6 (in the full version [GVV24]), and we thank Huck Bennett and Noah Stephens-Davidowitz for answering our questions about the BinaryBDD problem. We would also like to thank Stefan Tiegel for useful discussions about the performance of Lasso in the noiseless setting. AG was supported by the Ida M. Green MIT Office of Graduate Education Fellowship and by the grants of the third author. NV was supported by NSF fellowship DGE-2141064 and by the grants of the third author. VV was supported in part by DARPA under Agreement No. HR00112020023, NSF CNS-2154149, a grant from the MIT-IBM Watson AI, a grant from Analog Devices, a Microsoft Trustworthy AI grant, a Thornton Family Faculty Research Innovation Fellowship from MIT and a Simons Investigator Award. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Government or DARPA.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

Bab86. Babai, L.: On lovász'lattice reduction and the nearest lattice point problem. Combinatorica **6**, 1–13 (1986)

BDT24. Buhai, R.D., Ding, J., Tiegel, S.: Computational-statistical gaps for improper learning in sparse linear regression (2024). Personal Communication

BLP+13. Brakerski, Z., Langlois, A., Peikert, C., Regev, O., Stehlé, D.: Classical hardness of learning with errors. In: Boneh, D., Roughgarden, T., Feigenbaum, J., (eds.) Symposium on Theory of Computing Conference, STOC 2013, Palo Alto, CA, USA, June 1-4, 2013, pp. 575–584. ACM, (2013)

[8] We greatly thank Kiril Bangachev for turning the expected value statement into a high probability statement.

BRST21. Bruna, J., Regev, O., Song, M.J., Tang, Y.: Continuous LWE. In: Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing, pp. 694–707 (2021)
CDS98. Chen, S.S., Donoho, D.L., Saunders, M.A.: Saunders. Atomic decomposition by basis pursuit. SIAM J. Sci. Comput. **20**(1), 33–61 (1998)
CT07. Candes, E., Tao, T.: The Dantzig selector: statistical estimation when p is much larger than n. Ann. Stat. **35**(6), 2313–2351 (2007)
GKPV10. Goldwasser, S., Kalai, Y.T., Peikert, C.: Robustness of the learning with errors assumption. In: Chi-Chih Yao, A., (ed.) Innovations in Computer Science - ICS 2010, Tsinghua University, Beijing, China, January 5-7, 2010. Proceedings, pp. 230–240. Tsinghua University Press (2010)
GKZ21. Gamarnik, D., Kizildag, E.C., Zadik, I.: Inference in high-dimensional linear regression via lattice basis reduction and integer relation detection. IEEE Trans. Inf. Theory **67**(12), 8109–8139 (2021)
GV21. Gupte, A., Vaikuntanathan, V.: The fine-grained hardness of sparse linear regression. arXiv preprint arXiv:2106.03131 (2021)
GVV22. Gupte, A., Vafa, N., Vaikuntanathan, V.: Continuous LWE is as hard as lwe & applications to learning gaussian mixtures. In: 2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS), pp. 1162–1173. IEEE (2022)
GVV24. Gupte, A., Vafa, N., Vaikuntanathan, V.: Sparse linear regression and lattice problems. arXiv preprint arXiv:2402.14645 (2024)
KF15. Kirchner, P., Fouque, P.-A.: An improved BKW algorithm for LWE with applications to cryptography and lattices. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015. LNCS, vol. 9215, pp. 43–62. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-47989-6_3
KKMR21. Kelner, J.A., Koehler, F., Meka, R., Rohatgi, D.: On the power of preconditioning in sparse linear regression. CoRR, abs/2106.09207 (2021)
KKMR22. Kelner, J., Koehler, F., Meka, R., Rohatgi, D.: Lower bounds on randomly preconditioned lasso via robust sparse designs. Adv. Neural. Inf. Process. Syst. **35**, 24419–24431 (2022)
KKMR23. Kelner, J., Koehler, F., Meka, R., Rohatgi, D.: Feature adaptation for sparse linear regression. CoRR, abs/2305.16892, 2023
LLL82. Lenstra, A.K., Lenstra, H.W., Lovász, L.: Factoring polynomials with rational coefficients. Math. Annalen **261**, 515–534 (1982). https://doi.org/10.1007/BF01457454
LLM06. Liu, Y.-K., Lyubashevsky, V., Micciancio, D.: On bounded distance decoding for general lattices. In: Díaz, J., Jansen, K., Rolim, J.D.P., Zwick, U. (eds.) APPROX/RANDOM -2006. LNCS, vol. 4110, pp. 450–461. Springer, Heidelberg (2006). https://doi.org/10.1007/11830924_41
LM00. Laurent, B., Massart, P.: Adaptive estimation of a quadratic functional by model selection. Ann. Stat. **28**, 1302–1338 (2000)
Mic18. Micciancio, D.: On the hardness of learning with errors with binary secrets. Theory Comput. **14**(1), 1–17 (2018)
MP12. Micciancio, D., Peikert, C.: Trapdoors for Lattices: simpler, tighter, faster, smaller. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 700–718. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29011-4_41
NIS. NIST: Post-quantum cryptography standardization. https://csrc.nist.gov/Projects/Post-Quantum-Cryptography

NRWY12. Negahban, S.N., Ravikumar, P., Wainwright, M.J., Yu, B.: A unified framework for high-dimensional analysis of m-estimators with decomposable regularizers. Stat. Sci. **27**(4), 538–557 (2012). https://doi.org/10.1214/12-STS400

Pei09. Peikert, C.: Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In: Mitzenmacher, M., (ed.) Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009, pp. 333–342. ACM (2009)

Reg09. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. J. ACM, **56**(6), 34:1–34:40 (2009)

RV10. Rudelson, M., Vershynin, R.: Non-asymptotic theory of random matrices: extreme singular values. In: Proceedings of the International Congress of Mathematicians 2010 (ICM 2010) (In 4 Volumes) Vol. I: Plenary Lectures and Ceremonies Vols. II–IV: Invited Lectures, pp. 1576–1602. World Scientific (2010)

Sch87. Schnorr, C.-P.: A hierarchy of polynomial time lattice basis reduction algorithms. Theor. Comput. Sci. **53**, 201–224 (1987)

Tib96. Tibshirani, R.: Regression shrinkage and selection via the lasso. J. Roy. Stat. Soc.: Ser. B (Methodol.) **58**(1), 267–288 (1996)

ZSWB22. Zadik, I., Song, M.J., Wein, A.S., Bruna, J.: Lattice-based methods surpass sum-of-squares in clustering. In: Loh P.-L., Raginsky, M., (eds.) Conference on Learning Theory, 2-5 July 2022, London, UK, vol. 178 of Proceedings of Machine Learning Research, pp. 1247–1248. PMLR (2022)

ZWJ14. Zhang, Y., Wainwright, M.J., Jordan, M.I.: Lower bounds on the performance of polynomial-time algorithms for sparse linear regression. In: Conference on Learning Theory, pp. 921–948 (2014

Worst-Case to Average-Case Hardness of LWE: An Alternative Perspective

Aggarwal Divesh[1(✉)], Jin Ming Leong[2], and Veliche Alexandra[3]

[1] National University of Singapore, Singapore, Singapore
divesh@comp.nus.edu.sg
[2] Imperial College of London, London, U.K.
e0407679@u.nus.edu
[3] University of Michigan, Ann Arbor, MI, U.S.A.
aveliche@umich.edu

Abstract. In this work, we study the worst-case to average-case hardness of the Learning with Errors problem (LWE) under an alternative measure of hardness – the maximum success probability achievable by a probabilistic polynomial-time (PPT) algorithm. Previous works by Regev (STOC 2005), Peikert (STOC 2009), and Brakerski, Peikert, Langlois, Regev, Stehle (STOC 2013) give worst-case to average-case reductions from lattice problems to LWE, specifically from the approximate decision variant of the Shortest Vector Problem (GapSVP) and the Bounded Distance Decoding (BDD) problem. These reductions, however, are lossy in the sense that even the strongest assumption on the worst-case hardness of GapSVP or BDD implies only mild hardness of LWE. Our alternative perspective gives a much tighter reduction and strongly relates the hardness of LWE to that of BDD. In particular, we show that under a reasonable assumption about the success probability of solving BDD via a PPT algorithm, we obtain a nearly tight lower bound on the highest possible success probability for solving LWE via a PPT algorithm. Furthermore, we show a tight relationship between the best achievable success probability by any PPT algorithm for decision-LWE to that of search-LWE. Our results not only refine our understanding of the computational complexity of LWE, but also provide a useful framework for analyzing the practical security implications.

Keywords: LWE · BDD · success probability

1 Introduction

The Learning with Errors (LWE) problem has become one of the most important computational problems in post-quantum cryptography and computational complexity over the last two decades. Since Regev introduced this problem in 2005 [Reg09], the LWE problem has been used as the basis of a wide variety of cryptographic primitives, as well as a tool for proving hardness results in learning

E. Boyle and M. Mahmoody (Eds.): TCC 2024, LNCS 15365, pp. 308–336, 2025.
https://doi.org/10.1007/978-3-031-78017-2_11

theory [Reg06]. Formally, the LWE problem is defined as follows: The input consists of a uniformly random matrix $\mathbf{A} \sim \mathbb{Z}_p^{m \times n}$ and a vector $\mathbf{b} := \mathbf{As} + \mathbf{e} \in \mathbb{Z}_p^m$, where $\mathbf{s} \in \mathbb{Z}_p^n$ is a secret vector chosen uniformly at random from $\mathbb{Z}_p^n$ and $\mathbf{e} \in \mathbb{Z}_p^m$ is an error vector of small magnitude sampled according to a Gaussian distribution. The goal is to output $\mathbf{s}$. Here the positive integer p is called the modulus and n is the dimension. In his seminal work, Regev related LWE to worst-case lattice problems that form the foundation of lattice-based cryptography.

1.1 LWE and Computational Lattice Problems

Lattice Problems. A *lattice* $\mathcal{L}$ is a discrete additive subgroup of $\mathbb{R}^n$ that consists of all integer linear combinations of m linearly independent vectors $\mathcal{B} = \{\mathbf{b}_1, \mathbf{b}_2, \cdots, \mathbf{b}_m\} \subset \mathbb{R}^n$. Formally, it is defined as

$$\mathcal{L}(\mathcal{B}) := \left\{ \sum_{i=1}^{n} z_i \mathbf{b}_i \mid \forall i \in \{1, ..., n\}, z_i \in \mathbb{Z} \right\}.$$

We call m the rank, n the dimension of the ambient space, and $\mathcal{B}$ a basis of the lattice. A lattice can have many possible bases.

The two most important computational lattice problems are the *Shortest Vector Problem* (SVP) and the *Closest Vector Problem* (CVP). In SVP, one is given a basis for a lattice and asked to output a shortest non-zero lattice vector. We denote the length of a shortest non-zero vector of a lattice $\mathcal{L}$ by $\lambda_1(\mathcal{L})$. In the approximation variant of SVP, denoted by γ-SVP for some $\gamma > 1$, the goal is to output a nonzero lattice vector whose length is at most $\gamma\lambda_1(\mathcal{L})$. In CVP, one is given a target vector and basis for a lattice and asked to output a closest lattice vector to the target vector. Similarly, in its approximation variant γ-CVP, the goal is to output a lattice vector whose distance from the target vector is at most γ times the minimum distance between the target vector and the lattice. There exists a polynomial-time reduction from SVP to CVP, which preserves the dimension, rank, and approximation factor [GMSS99].

A closely related problem to CVP is the *Bounded Distance Decoding* (BDD) problem, denoted by BDD_α for some $\alpha < \frac{1}{2}$. This is a promise problem in which the goal is to solve CVP under the promise that the distance of the target from the lattice is at most $\alpha\lambda_1(\mathcal{L})$. Note that, by the triangle inequality, this promise ensures that the closest vector to the target is unique. In our work, we only consider length and distance in the standard Euclidean (ℓ_2) norm.

Computational lattice problems are crucial because of their association with lattice-based cryptography. Specifically, the security of numerous cryptographic systems such as [Ajt96, MR04, Reg06, MR09, Reg09, Gen09, BV14] relies on the complexity of approximately solving lattice problems to within a polynomial factor. Aside from cryptosystem design, since the 1980s, solvers for lattice problems have found applications in cryptanalytic tools [Sha85, Bri83, LO85], algorithmic number theory [LLL82], and convex optimization [Kan87, FT87].

Algorithms for Lattice Problems. Algorithms for CVP and SVP have been designed and studied extensively for decades. Kannan proposed an enumeration algorithm [Kan87] for CVP and hence for all lattice problems, with a time complexity of $n^{O(n)}$ and space requirement of poly(n). Micciancio and Voulgaris introduced a deterministic algorithm for CVP with a time complexity of $2^{2n+o(n)}$ and space requirement of $2^{n+o(n)}$ [MV13]. A few years later, Aggarwal, Dadush, Regev, and Stephens-Davidowitz [ADRS14, ADRS15] presented the current fastest known algorithm for SVP and CVP, which has a time and space complexity of $2^{n+o(n)}$. The best-known and proven runtime for an approximation factor $\gamma = n^c$ is approximately $2^{n/(c+1)}$ for constant $c \geq 0$. For the current state of the art, we refer the reader to [ALS20].

Hardness of Lattice Problems. Both γ-SVP and γ-CVP are known to be NP-hard for nearly-polynomial approximation $n^{c/\log\log n}$ for some constant $c > 0$ [vEB81, DKRS03, Din02, Kho05, HR18, Mic12]. Through a series of works, Aggarwal, Bennett, Golovnev, and Stephens-Davidowitz [BGS17, ASD18, ABGSD21], demonstrated that approximating CVP and SVP to a factor γ slightly greater than 1 is not achievable in time $2^{o(n)}$ under variants of the Exponential Time Hypothesis.

Worst-case to Average-case Reduction for LWE. The best known algorithm that solves LWE for dimension n and modulus p runs in time $p^{O(n/\log n)}$ [BKW00]. The decision variant of LWE is the one most directly related to the security of lattice-based cryptography. In decision-LWE, the goal is to distinguish between an LWE instance as described above and a uniform sample from $\mathbb{Z}_p^{m\times(n+1)}$. Regev [Reg09] gave a polynomial-time reduction from BDD to LWE. Additionally, Regev gave a quantum polynomial-time reduction from a decision variant of γ-SVP, known as GapSVP$_\gamma$, to BDD for γ polynomial in the dimension of the lattice. Peikert [Pei09] improved this result to a classical reduction from GapSVP$_\gamma$ to LWE, albeit with the modulus p becoming exponential in the dimension n. Later, Brakerski et al. [BLP+13] gave a reduction from LWE with dimension n and modulus p exponential in n to LWE with dimension n^2 and modulus p polynomial in n, thus allowing the modulus to shrink from exponential to polynomial.

Overall, these results give us a polynomial-time reduction from lattice problems (GapSVP, BDD) in dimension n to LWE in dimension n^2 with modulus p polynomial in n. This means that even if we assume that the currently best known algorithms for BDD or GapSVP are the best possible, this reduction only says that LWE in dimension n cannot be solved faster than in $2^{\Omega(\sqrt{n})}$ time. This is a much worse lower bound than one would expect based on the state-of-the-art algorithms for LWE [BKW00]. This leads us to the following natural question.

Question 1. Is there a tight reduction from worst-case lattice problems (such as BDD) to LWE that gives a tighter lower bound on the runtime for solving LWE?

1.2 A Novel Perspective on Computational Hardness

In cryptography, security models often assume that all possible adversaries are computationally bounded, based on the state-of-the-art capabilities of modern computers. Often, when we declare that a cryptographic scheme is 256-bit secure, we intuitively understand that the fastest algorithm for successfully breaking the cryptosystem runs in 2^{256} units of time. What we typically require, however, is that any algorithm that succeeds in attacking the cryptosystem with probability more than 2^{-256} cannot do so in a "reasonable" amount of time.

Unpredictability Entropy. Motivated by this discrepancy, Aggarwal and Maurer [AM11] proposed a different perspective on studying the complexity of a computational search problem. They introduced the concept of *unpredictability entropy* for a computational problem, defined as follows. If p is the maximum success probability of a probabilistic polynomial-time (PPT) algorithm that solves the problem, the unpredictability entropy of the problem is $\log_2(\frac{1}{p})$.

Two closely related properties of a search problem also studied in [AM11] are witness compression and oracle complexity. A search problem $\mathcal{P}$ is said to have *witness compression* w if there is a PPT reduction from $\mathcal{P}$ to another search problem $\mathcal{Q}$ such that the problem $\mathcal{Q}$ has a solution/witness of length w. The *oracle complexity* of the problem $\mathcal{P}$ is defined as the number of arbitrary YES/NO questions needed to get a solution to the search problem, i.e. find a witness.

It was shown in [AM11] that unpredictability entropy, witness compression, and oracle complexity of a computational problem are equal, up to lower order additive terms. Notice that these quantities are all indicative of the number of bits in which the hardness of a computational problem can be captured.

The authors of [AM11] also gave a straightforward polynomial-time algorithm for both SVP and CVP, that achieves a success probability of $2^{-n^2/4-o(n^2)}$, showing that both these problems have unpredictability entropy/witness compression/oracle complexity $n^2/4 + o(n^2)$. These algorithms are straightforward adaptations of the LLL algorithm and Babai's nearest plane algorithm [LLL82, Bab86]. If we replace [LLL82] with the slide reduction algorithm [GN08, ALNSD20] with block length $O(\log n)$, this still runs in polynomial time and reduces the search space, giving a success probability $2^{-\Theta(n^2/\log n)}$. Despite the various algorithmic techniques available for solving lattice problems, none of these methods appear to improve this further if we are restricted to PPT algorithms, even if we consider approximation variants of the lattice problems with approximation factor γ polynomial in the lattice dimension n. Additionally, the close relationship between BDD_α and γ-SVP for $\frac{1}{\alpha}$ and γ both polynomial in n [LM09], suggests that it is unlikely for a polynomial-time algorithm for BDD to do much better than current algorithms. With this in mind, it is reasonable to conjecture the following.

Conjecture 1. For any constants $c, c' > 0$, there exists $\kappa = \kappa(c, c') > 0$ such that no algorithm can solve $\mathsf{BDD}_{1/\gamma}$, γ-SVP, and γ-CVP on an arbitrary lattice for

approximation factor $\gamma = n^c$ in time $n^{c'}$ with success probability better than $2^{-n^2/\kappa \log n}$.

It is easy to see that any algorithm for solving LWE in polynomial time for modulus p and dimension n has success probability at least p^{-n}. This can be obtained by guessing the secret $\mathbf{s}$ uniformly at random and then checking whether $\mathbf{b} - \mathbf{As}$ is small. We ask the following natural question, which is a novel perspective on the worst-case to average-case reductions for LWE.

Question 2. Assuming Conjecture 1, is there a lower bound close to $p^{-\Omega(n)}$ on the success probability of solving LWE via a polynomial-time algorithm?

In this work, we answer this question in the affirmative. Since the security of cryptosystems is based on the hardness of decision problems, we need to adapt the question above and formulate the measure of hardness of a decision problem in terms of the success probability of the best efficient (i.e. PPT) algorithm.

One-Sided Error PPT Algorithms. This question has been well studied particularly in the context of NP-hard problems, for which we expect any PPT algorithm to be able to distinguish only with a small advantage. Some previous works, (such as [PP10], and the references therein) have explored the realm of *one-sided error* probabilistic polynomial-time (OPP) algorithms for NP-hard decision problems. This relies on the assumption that when the algorithm is presented with a NO instance, it consistently outputs NO, while for a YES instance, the algorithm outputs YES with a small success probability α.

However, for decision problems like decision-LWE whose input is chosen according to a distribution, we cannot hope to output NO with probability 1 even given a NO instance. This is because a NO instance for this problem is just a random element from $\mathbb{Z}_q^{m \times (n+1)}$ that will look like a YES instance with non-zero probability. Thus, it is reasonable to adapt the notion of OPP algorithms to have two parameters α, β such that $\alpha \gg \beta$, and the algorithm outputs YES with probability at least α, when given a YES instance, and with probability at most β when given a NO instance. We call such an algorithm an (α, β)-*solver* for decision LWE. Using this notation, we ask the following natural question.

Question 3. Assuming that there is no PPT algorithm that succeeds in solving search-LWE with probability α, can we prove that there is no (α', β')-solver for the corresponding decision-LWE problem with $\alpha' \approx \alpha$ and $\beta' \ll \alpha'$.

We also answer this question in the affirmative.

1.3 Our Contributions

In this paper, we show that if no PPT algorithm can solve BDD on a lattice of rank n with success probability greater than $2^{-O(n^2)/\log n}$, then no PPT algorithm can solve search-LWE in n dimensions with success probability greater than $2^{-O(n^2)/\log n}$. Here n is the dimension of the lattice even if we restrict the secret to be a binary vector. Informally, our first main result is the following.

Theorem 1. *(informal) If no PPT algorithm can solve* BDD_γ *for gap* $\gamma \in (0, \frac{1}{2})$ *with success probability greater than* $2^{-n^2/\kappa \log n} - 4$ *for some* $\kappa > 0$*, then no PPT algorithm can solve search-*LWE *for binary secret and modulus* p *polynomial in the dimension* n *with success probability* $2^{-n^2/\kappa \log n}$*.*

Note that the above statement can easily be extended to having the secret chosen uniformly at random from $\mathbb{Z}_p^n$ using a standard randomization of the secret. We show this explicitly in Sect. 3.3.

We emphasize here that while our reductions are adaptations of similar reductions in the literature, adapting these reductions to our setting required great care, since the number of oracle calls made in the reductions is crucial. In particular, for a reduction from problem $\mathcal{P}$ to problem $\mathcal{Q}$ that makes k calls to a solver for problem $\mathcal{Q}$, an upper bound of δ on the success probability for solving problem $\mathcal{P}$ in polynomial time would imply an upper bound of $\delta^{1/k}$ on the success probability for solving problem $\mathcal{Q}$ in polynomial time. So for our reductions, we needed to adapt known reductions, which make polynomially many oracle calls, into reductions that make only one call to the oracle and then guess successfully with a small probability. These reductions with a single oracle call are known as *one-shot reductions*. This approach enables us to obtain meaningful bounds on the success probability of polynomial-time reductions. We also remark that since Regev's quantum reduction does not blow up the dimension, our results do not have any novel implication in the quantum setting.

Our second main contribution is concretely relating the hardness of solving decision-LWE to that of solving search-LWE using our new framework. In particular, we show that if no algorithm can solve search-LWE on a lattice of rank n with modulus p in expected polynomial time with success probability close to α, then there is no PPT algorithm that can solve decision-LWE for the same dimension and modulus and answers correctly with probability close to α, outputs $\perp$ with probability $1 - \alpha$, and answers incorrectly with the remaining tiny probability. This relies on the assumption that β is large and close to 1, which requires the oracle $\mathcal{B}$ to be correct with high probability when it does not output $\perp$. Intuitively, this means that $\mathcal{B}$ admits defeat by outputting $\perp$ far more often than it guesses the answer incorrectly. Using this framework, we can informally state our second main result as follows.

Theorem 2. *(informal) If no algorithm can solve search-*LWE *for modulus* p *polynomial in the dimension* n *with success probability* α *in expected polynomial time, then no PPT algorithm can solve decision-*LWE *for the same modulus* p *and dimension* n *that outputs a correct answer with probability* α *and outputs* $\perp$ *with probability* $1 - \alpha$*.*

To prove our second main result, we use a result by Levin [Lev12] This result is an improvement of the original Goldreich-Levin Theorem in [GL89] which gives a tight relationship between the success probability of finding a hard-core bit and that of inverting the corresponding one-way function. In our work, we rigorously prove Levin's result and generalise it from binary $\{0, 1\}$ to $\mathbb{Z}_p$ for all but a few values of p. This required considerable care and can easily find

applications elsewhere, so we consider it to be a contribution of independent interest.

Note that the statement of Theorem 2 is in terms of *expected* polynomial time, which is a crucial aspect of the Goldreich-Levin Theorem used in our reduction. Because the runtime is polynomial only in expectation, we cannot directly combine this result with that of Theorem 1.

We remark here that while the techniques used to prove our second main result are similar to those used in [MM11], we do not require that the probability α of the decision-LWE oracle answering correctly is non-negligible. Instead, we only require that α is sufficiently larger than the probability δ of answering incorrectly. In particular, this means that both α and δ can be exponentially small. In our work, we also explicitly quantify the loss in success probability for each of our reductions.

1.4 Paper Organization

We give the notation and mathematical background needed for our results and formally define the computational problems discussed throughout the paper in Sect. 2. In Sect. 3, we prove our first main result, where we use techniques from [Reg09] with new tweaks. In Sect. 4, we prove our second main result, where we reduce the hardness of solving search-LWE to the hardness of solving decision-LWE using our new framework [MW18]. We conclude with future directions and open problems in Sect. 5.

2 Preliminaries

Let $\mathbb{T} := \mathbb{R}/\mathbb{Z}$ denote the additive group of real numbers modulo the integers, i.e. the interval $[0, 1)$ with addition modulo 1. $\mathbb{R}_+$ and $\mathbb{Z}_+$ denote the positive real numbers and positive integers, respectively, which do not include zero. Let p be any positive integer (not necessarily prime). $\mathbb{Z}_p := \mathbb{Z}/p\mathbb{Z}$ denotes the ring of integers $\{0, 1, \ldots, p-1\}$ where addition and multiplication are performed modulo p. We implicitly identify $\mathbb{Z}_p$ with its natural embedding in $\mathbb{Z}$ whenever relevant. For any $n \in \mathbb{Z}_+$, we denote $[n] := \{1, ..., n\}$ to be all the integers between 1 and n, inclusive. We use $\langle \cdot, \cdot \rangle$ to denote the standard dot product, so $\langle \mathbf{x}, \mathbf{y} \rangle = \mathbf{x}^{\mathrm{T}}\mathbf{y}$ for column vectors $\mathbf{x}, \mathbf{y}$ where addition and multiplication are performed according to the domain. We use lowercase boldface letters (such as $\mathbf{v}$), to denote vectors, uppercase boldface letters (such as $\mathbf{B}$) to denote matrices, and calligraphic uppercase boldface letters (such as $\mathcal{A}$) to denote algorithms. We use $\|\cdot\| = \|\cdot\|_2$ to denote the Euclidean norm. Throughout this paper, all norms are assumed to be Euclidean unless specified otherwise. We say that an algorithm is *efficient* if it runs in time polynomial in the size of the input, and use the terms "efficient" and "polynomial-time" interchangeably throughout.

We will need the following standard lemma.

Lemma 1. *Let $Y_1, \ldots, Y_t$ be pairwise independent Bernoulli random variables where $\Pr[Y_i = 1] = p$ for $1 \leq i \leq t$. Then for any $c > 0$,*

$$\Pr\left[\left|\sum_{i=1}^{t} Y_i - tp\right| \leq ctp\right] \geq 1 - \frac{1}{c^2 tp} .$$

Proof. Let $Y = Y_1 + \cdots + Y_t$. The expected value of Y is $\mathbb{E}[Y] = tp$ and the variance of Y is $\mathrm{Var}[Y] = tp(1-p)$. Then by the Chebyshev inequality, we have

$$\Pr[|Y - tp| \geq ctp] \leq \frac{tp(1-p)}{c^2 t^2 p^2} = \frac{1-p}{c^2 tp} \leq \frac{1}{c^2 tp} .$$

□

2.1 Learning with Errors

Definition 1 (LWE **Distribution).** *Let ϕ be a probability density function on $\mathbb{T}$, and $\mathbf{s} \in \mathbb{Z}_p^n$ denote the unknown secret vector. The* Learning with Errors (LWE) distribution *$A_{\mathbf{s},\phi}$ is the distribution over $\mathbb{Z}_p^n \times \mathbb{T}$ obtained by choosing $\mathbf{a} \in \mathbb{Z}_p^n$ uniformly at random and $e \in \mathbb{T}$ according to ϕ, then outputting $(\mathbf{a}, \frac{1}{p}\langle \mathbf{a}, \mathbf{s} \rangle + e)$.*

The standard Learning with Errors problem has both search and decision variants, defined as follows.

Definition 2 (Search-LWE**).** *The search variant of the Learning with Errors problem,* search-LWE$_{n,p,\phi}$*, is defined as: given a polynomial number of samples from the distribution $A_{\mathbf{s},\phi}$, recover the secret $\mathbf{s} \in \mathbb{Z}_p^n$.*

Definition 3 (Decision-LWE**).** *The decision variant of the Learning with Errors problem,* decision-LWE$_{n,p,\phi}$*, is defined as: given a polynomial number of samples either from the distribution $A_{\mathbf{s},\phi}$ or independent and uniformly distributed samples from $\mathbb{Z}_p^n \times \mathbb{T}$, output*

- *YES if the samples are from the* LWE *distribution $A_{\mathbf{s},\phi}$, or*
- *NO if the samples are uniformly random over $\mathbb{Z}_p^n \times \mathbb{T}$.*

Definition 4 (Binary-LWE**).** *The Binary Learning with Errors problem,* binLWE$_{n,p,\phi}$*, is the search-*LWE$_{n,p,\phi}$ *problem with the restriction that the secret $\mathbf{s}$ is uniform over $\{0,1\}^n$.*

2.2 Lattices

Definition 5 (Lattice). *Let $\mathcal{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_m\} \subseteq \mathbb{R}^n$ be a set of linearly independent vectors. The* lattice *$\mathcal{L} = \mathcal{L}(\mathcal{B})$ generated by $\mathcal{B}$ is the set of vectors spanned by $\mathcal{B}$ over $\mathbb{Z}$, i.e.*

$$\mathcal{L}(\mathcal{B}) := \left\{ \sum_{i=1}^{m} z_i \mathbf{b}_i \;\middle|\; \mathbf{z} = (z_1, \ldots, z_n)^T \in \mathbb{Z}^n \right\} \subset \mathbb{R}^n .$$

The basis $\mathcal{B}$ *is usually expressed as a matrix* $\mathbf{B}$ *whose columns are the vectors of* $\mathcal{B}$ *and we write* $\mathcal{L}(\mathbf{B}) := \mathcal{L}(\mathcal{B})$*. Here* m *is the* rank *of the lattice as a free* $\mathbb{Z}$*-module, and* n *is the* dimension *of the ambient space.*

Throughout this paper, we assume that the lattices are *full-rank*, meaning that $n = m$.

Definition 6 (Determinant). *The* determinant *of lattice* $\mathcal{L}$ *generated by* $\mathbf{B} = (\mathbf{b}_1, \ldots, \mathbf{b}_m) \in \mathbb{R}^{n \times m}$ *is*

$$\det(\mathcal{L}) := \sqrt{\det(\mathbf{B}^T\mathbf{B})}.$$

Definition 7 (Dual Lattice). *The* dual lattice *of* $\mathcal{L}$ *is*

$$\mathcal{L}^* := \{\mathbf{x} \in \mathbb{R}^n \mid \forall\ \mathbf{v} \in \mathcal{L},\ \langle \mathbf{v}, \mathbf{x} \rangle \in \mathbb{Z}\}.$$

Given a basis matrix $\mathbf{B} \in \mathbb{R}^{n \times m}$ *of a lattice* $\mathcal{L}$*, we write* $\mathbf{B}^*$ *to denote the corresponding basis matrix for* $\mathcal{L}^*$*. This satisfies* $(\mathbf{B}^*)^T\mathbf{B} = \mathbf{I}_m$.

Definition 8 (Successive Minima). *For any* $k \in \mathbb{Z}_+$*, the* k-th successive minimum *of* $\mathcal{L}$ *(in the Euclidean norm) is*

$$\lambda_k(\mathcal{L}) := \inf\{r \in \mathbb{R} \mid B(\mathbf{0}, r) \text{ contains } k \text{ linearly independent vectors}\},$$

where $B(\mathbf{0}, r)$ *denotes the ball of radius* r *centered at the origin. In particular,* $\lambda_1(\mathcal{L})$ *is the length of any shortest nonzero vector in* $\mathcal{L}$*.*

Definition 9 (Unique Closest Lattice Vector). *For any vector* $\mathbf{v} \in \mathbb{R}^n$ *whose distance from the lattice* $\mathcal{L}$ *is less than* $\lambda_1(\mathcal{L})/2$*, there is a unique closest lattice vector to* $\mathbf{v}$*, which we denote by* $\kappa_{\mathcal{L}}(\mathbf{v})$*.*

2.3 Probability and Gaussians

Throughout this paper, we frequently use the standard normal distribution over the real numbers. We use the standard notation $N(\mu, \sigma^2)$ to denote the normal distribution with mean μ and variance σ^2.

Definition 10 (Statistical Distance). *Let* ϕ_1 *and* ϕ_2 *be probability measures on the space* $(X, \mathcal{F})$*, where* X *is the set of outcomes and* $\mathcal{F}$ *is the collection of events. The* statistical distance *(a.k.a total variation distance) between* ϕ_1 *and* ϕ_2 *is*

$$\Delta(\phi_1, \phi_2) := \sup_{A \in \mathcal{F}} \{|\phi_1(A) - \phi_2(A)|\}.$$

In particular, when $X = \mathbb{R}^n$*,*

$$\Delta(\phi_1, \phi_2) = \frac{1}{2} \int_{\mathbb{R}^n} |\phi_1(\mathbf{x}) - \phi_2(\mathbf{x})| d\mathbf{x}.$$

If X and Y are random variables with distributions ϕ_1 and ϕ_2, respectively, we define $\Delta(X,Y) := \Delta(\phi_1, \phi_2)$. It is immediate that statistical distance satisfies the triangle inequality. Another important property is that it does not increase under the application of any (possibly random) function f [Vad12], i.e.

$$\Delta(f(X), f(Y)) \leq \Delta(X,Y). \qquad (*)$$

This means that for any algorithm $\mathcal{A}$, the success probability of $\mathcal{A}$ on X differs from the success probability of $\mathcal{A}$ on Y by at most $\Delta(X,Y)$.

Definition 11 (Gaussian Function). *The* Gaussian function *of width $s \in \mathbb{R}_+$ is $\rho_s : \mathbb{R}^n \to \mathbb{R}$, given by*

$$\rho_s(\mathbf{x}) := e^{-\pi \left\| \frac{\mathbf{x}}{s} \right\|^2}.$$

For any countable subset $A \subseteq \mathbb{R}^n$, we write $\rho_s(A) := \sum_{\mathbf{x} \in A} \rho_s(\mathbf{x})$. Furthermore, we denote $\rho := \rho_1$. For any $\mathbf{y} \in \mathbb{R}^n$, we define $\rho_{s,\mathbf{y}}(\mathbf{x}) := \rho_s(\mathbf{x} - \mathbf{y})$.

Note that $\int_{\mathbb{R}^n} \rho_s(\mathbf{x}) d\mathbf{x} = s^n$. Hence

$$\nu_s := \frac{\rho_s}{s^n}$$

is a probability density function on $\mathbb{R}^n$, which we call a continuous Gaussian of width s. Similarly, we write $\nu := \nu_1$. Since a sample from ν_s can be generated by taking n independent samples from the 1-dimensional Gaussian distribution, we assume that we can sample efficiently from ν_s.

Definition 12 (Discrete Gaussian). *For any countable subset A for which $\rho_s(A)$ converges, $D_{A,s} : A \to \mathbb{R}_+$ is the* discrete Gaussian distribution *on A defined by*

$$D_{A,s}(\mathbf{x}) := \frac{\rho_s(\mathbf{x})}{\rho_s(A)}.$$

Definition 13 (Distribution Ψ_γ). *For any $\gamma \in \mathbb{R}_+$, define the distribution $\Psi_\gamma : \mathbb{T} \to \mathbb{R}_+$ by*

$$\Psi_\gamma(r) := \sum_{k \in \mathbb{Z}} \frac{1}{\gamma} e^{-\pi \left(\frac{r-k}{\gamma} \right)^2}.$$

In other words, if X is distributed according to Ψ_γ and $Z \sim N(0, \frac{\gamma^2}{2\pi})$, then X is the image of Z modulo 1.

An important property used in [Reg09] is the fact that Ψ_β does not change much under a small change in the parameter β. In [Reg09], it was shown that the statistical distance between Ψ_β and another distribution Ψ_α, such that β is not too far from α is bounded by a scaling of the ratio $\frac{\beta}{\alpha}$. For our reduction, we need a much tighter bound, so we instead use the ratio between the probability density functions corresponding to Ψ_α and Ψ_β. Formally, we show the following.

Lemma 2. *Let $\alpha \in \mathbb{R}_+$ and $\beta := \alpha(1+\varepsilon)$ for some $\varepsilon > 0$. Denote the probability density functions of distributions Ψ_α and Ψ_β by g_α and g_β, respectively. Then their ratio satisfies*

$$\frac{g_\alpha(x)}{g_\beta(x)} \leq 1 + \varepsilon = \frac{\beta}{\alpha}.$$

for any $x \in \mathbb{R}$.

Proof. Suppose X and X' are distributed according to Ψ_α and Ψ_β, respectively. By definition, X is the image of some Z with distribution $N(0, \frac{\alpha^2}{2\pi})$ modulo 1, and similarly, X' is the image of some Z' with distribution $N(0, \frac{\beta^2}{2\pi})$ modulo 1. Then the ratio of the probability density functions f_α and f_β is the same as the ratio of the probability density functions of Z and Z' modulo 1. By this we obtain

$$\begin{aligned}\frac{g_\alpha(x)}{g_\beta(x)} &= \frac{\frac{1}{\alpha}e^{-\pi(\frac{x}{\alpha})^2}}{\frac{1}{\beta}e^{-\pi(\frac{x}{\beta})^2}} = \frac{\beta}{\alpha}e^{-\pi x^2 \frac{1}{\alpha^2} + \pi x^2 \frac{1}{\beta^2}} = \frac{\alpha(1+\varepsilon)}{\alpha}e^{-\pi x^2\left(\frac{1}{\alpha^2} - \frac{1}{\alpha^2(1+\varepsilon)^2}\right)} \\ &= (1+\varepsilon)e^{-\pi\frac{x^2}{\alpha^2}\left(1-\frac{1}{(1+\varepsilon)^2}\right)} \leq 1+\varepsilon.\end{aligned}$$

The last inequality follows from the fact that $f(x) := e^{-\pi x^2 \frac{1}{\alpha^2}\left(1-\frac{1}{(1+\varepsilon)^2}\right)}$ is a scaling of the Gaussian curve, so $f(x) \leq 1$ for any value of x. □

Now we prove a multiplicative analog of (∗) for this ratio of probability density functions.

Lemma 3. *Let X and Y be continuous random variables in $\mathbb{R}$ with probability density functions g_X and g_Y, respectively. Suppose that for some fixed $\delta > 0$, their ratio satisfies*

$$\frac{g_X(x)}{g_Y(x)} \leq \delta$$

for all x. Then for any (invertible) function $f : U \to V$ and set $S \subseteq V$,

$$\frac{\Pr[f(X) \in S]}{\Pr[f(Y) \in S]} \leq \delta.$$

Proof. Consider the set $T^* := \left\{u \in U \mid \frac{\Pr[X=u]}{\Pr[Y=u]} > 0\right\}$. This maximises the ratio $\frac{\Pr[X\in T]}{\Pr[Y\in T]}$ over all $T \subseteq U$. This enables us to write

$$\frac{\Pr[f(X) \in S]}{\Pr[f(Y) \in S]} = \frac{\Pr[X \in f^{-1}(S)]}{\Pr[Y \in f^{-1}(S)]} \leq \max_{T\subseteq U}\left\{\frac{\Pr[X \in T]}{\Pr[Y \in T]}\right\} = \frac{\Pr[X \in T^*]}{\Pr[Y \in T^*]}.$$

By definition of the probability density function, we have

$$\frac{\Pr[X \in T^*]}{\Pr[Y \in T^*]} = \frac{\int_{T^*} g_X(x)dx}{\int_{T^*} g_Y(x)dx} \leq \frac{\delta \int_{T^*} g_Y(x)dx}{\int_{T^*} g_Y(x)dx} = \delta.$$

□

We remark that the statement can easily be extended to any randomised function f. This means that for any algorithm $\mathcal{A}$, the success probability of $\mathcal{A}$ on X differs from the success probability of $\mathcal{A}$ on Y by at most a multiplicative factor of $\frac{1}{\delta}$. We will use Lemmas 2 and 3 in Sect. 3.2 for the reduction from generised-LWE to standard search-LWE.

2.4 The Smoothing Parameter

For any lattice $\mathcal{L}$, one can show that $\rho_t(\mathcal{L})$ converges for all $t > 0$. In particular, the map $s \mapsto \rho_{1/s}(\mathcal{L} \setminus \{\mathbf{0}\})$ is a strictly decreasing continuous map on $\mathbb{R}_+$, that satisfies $\lim_{s\to\infty}\{\rho_{1/s}(\mathcal{L} \setminus \{\mathbf{0}\})\} = 0$ and $\lim_{s\to 0}\{\rho_{1/s}(\mathcal{L} \setminus \{\mathbf{0}\})\} = \infty$. This enables us to define the following parameter.

Definition 14 (Smoothing Parameter). *Let $\mathcal{L} \subset \mathbb{R}^n$ be a lattice and $\varepsilon > 0$. The* smoothing parameter *of $\mathcal{L}$ with respect to ε is*

$$\eta_\varepsilon(\mathcal{L}) := \inf\{s \in \mathbb{R}_+ \mid \rho_{\frac{1}{s}}(\mathcal{L}^* \setminus \{\mathbf{0}\}) \leq \varepsilon\}.$$

By the above observation on the map $s \mapsto \rho_{1/s}(\mathcal{L} \setminus \{\mathbf{0}\})$, the infimum in the definition above can be achieved with equality. In fact, $s \mapsto \rho_{1/s}(L \setminus \{\mathbf{0}\})$ is a bijection from $\mathbb{R}_+$ to $\mathbb{R}_+$ with inverse $\varepsilon \mapsto \eta_\varepsilon(\mathcal{L})$.

Observe that, using the properties of the Gaussian function and the fact that $(p\,\mathcal{L})^* = p^{-1}\,\mathcal{L}^*$, any scaling of the smoothing parameter can be rewritten as

$$\begin{aligned} p \cdot \eta_\varepsilon(\mathcal{L}) &= \inf\{ps \in \mathbb{R}_+ \mid \rho_{\frac{1}{s}}(\mathcal{L}^* \setminus \{\mathbf{0}\}) \leq \varepsilon\} \\ &= \inf\{s' \in \mathbb{R}_+ \mid \rho_{\frac{p}{s'}}(\mathcal{L}^* \setminus \{\mathbf{0}\}) \leq \varepsilon\} \\ &= \inf\{s' \in \mathbb{R}_+ \mid \rho_{\frac{1}{s'}}(p^{-1}\,\mathcal{L}^* \setminus \{\mathbf{0}\}) \leq \varepsilon\} \\ &= \eta_\varepsilon(p\,\mathcal{L}). \end{aligned}$$

The following upper and lower bounds on the smoothing parameter will be used in our reduction.

Lemma 4. *(Claim 2.13 from [Reg09]) For any n-dimensional lattice $\mathcal{L}$ and $\varepsilon \in \mathbb{R}_+$ we have*

$$\eta_\varepsilon(\mathcal{L}) \geq \sqrt{\frac{1}{\pi}\ln\left(\tfrac{1}{\varepsilon}\right)} \cdot \frac{1}{\lambda_1(\mathcal{L}^*)} \geq \sqrt{\frac{1}{\pi}\ln\left(\tfrac{1}{\varepsilon}\right)} \cdot \frac{\lambda_n(\mathcal{L})}{n}.$$

Lemma 5. *(Lemma 3.1 from [GPV08], adapted) For any n-dimensional lattice $\mathcal{L}$ with basis $\mathbf{B} = \{\mathbf{b}_1, ..., \mathbf{b}_n\}$ and $\varepsilon \in \mathbb{R}_+$ we have*

$$\eta_\varepsilon(\mathcal{L}) \leq \max_{i\in[n]}\{\|\mathbf{b}_i\|\} \cdot \sqrt{\frac{1}{\pi}\ln\left(2n\left(1+\frac{1}{\varepsilon}\right)\right)}.$$

We remark that the original Lemma 3.1 in [GPV08] is tighter, as the maximum is over the Gram-Schmidt orthogonolization of the basis vectors, $\tilde{\mathbf{b}}_1, ..., \tilde{\mathbf{b}}_m$, which satisfy $\|\tilde{\mathbf{b}}_i\| \leq \|\mathbf{b}_i\|$ for all $i \in [n]$. The original statement takes the minimum of this maximum norm over all possible bases $\mathbf{B}$ of the lattice.

The following is an elementary bound on the shortest vector length in the dual lattice.

Lemma 6. *(Theorem 3.2 from [Cai98], adapted) For any n-dimensional lattice $\mathcal{L}$ with basis $\mathbf{B} = \{\mathbf{b}_1, ..., \mathbf{b}_n\}$,*

$$\frac{1}{\lambda_1(\mathcal{L}^*)} \leq \max_{i \in [n]} \{\|\mathbf{b}_i\|\}.$$

As in the previous lemma, the statement above is weaker than the original statement in [Cai98], as it uses the weaker bound given by $\|\tilde{\mathbf{b}}_i\| \leq \|\mathbf{b}_i\|$ for all $i \in [n]$, instead of the smallest maximum length of the Gram-Schmidt orthogonolization of the basis vectors taken over all possible bases of the lattice. For our purposes, it suffices to take the weaker versions of these bounds stated in the lemmas above.

2.5 Computational Lattice Problems

Definition 15 (SVP). *Let $\gamma \geq 1$. The* γ-approximate Shortest Vector Problem *in the Euclidean norm,* GapSVP_γ*, is the decision problem defined as: given an instance $(\mathbf{B}, d)$ consisting of a basis matrix $\mathbf{B}$ of a rank-n lattice $\mathcal{L}$ and distance parameter $d > 0$, output*

- *YES if $\lambda_1(\mathcal{L}) \leq d$, or*
- *NO if $\lambda_1(\mathcal{L}) \geq \gamma d$.*

Definition 16 (BDD). *The* Bounded Distance Decoding problem, BDD_α*, parameterized by an approximation factor $\alpha \in (0, \frac{1}{2})$ is the search problem defined as: given a basis matrix $\mathbf{B}$ of a rank-n lattice $\mathcal{L}$ and a target vector $\mathbf{v} \in \mathbb{R}^n$ with the promise that $\mathrm{dist}(\mathbf{v}, \mathcal{L}) < \alpha \cdot \lambda_1(\mathcal{L})$, find a lattice vector closest to $\mathbf{v}$, i.e. an $\mathbf{x} \in \mathcal{L}$ such that $\|\mathbf{v} - \mathbf{x}\| < \alpha \cdot \lambda_1(\mathcal{L})$.*

Definition 17 (mod-BDD). *Let $p \in \mathbb{Z}_+$ and $\alpha \in (0, \frac{1}{2})$. The* Modulo-$p$ Bounded Distance Decoding problem, $\mathsf{BDD}_{\alpha,p}$*, is the search problem defined as: given a basis matrix $\mathbf{B}$ of a rank-n lattice $\mathcal{L}$ and a target vector $\mathbf{v} \in \mathbb{R}^n$ with the promise that $\mathrm{dist}(\mathbf{v}, \mathcal{L}) < \alpha \cdot \lambda_1(\mathcal{L})$, find the coefficient vector of a lattice vector closest to $\mathbf{v}$ modulo p. i.e. if $\mathbf{x} \in \mathcal{L}$ is closest to $\mathbf{v}$, then the expected output is $\mathbf{B}^{-1}\mathbf{x} \pmod p \in \mathbb{Z}_p^n$.*

3 BDD to Search-LWE

We now formally state and prove our first main result.

Theorem 3. (BDD $\rightarrow$ search-LWE) *Let $\alpha = \alpha(n) \in (0,1)$ and $\gamma \in (0, \frac{1}{2})$. Suppose there exists a polynomial-time algorithm $\mathcal{B}$ that solves $\mathrm{LWE}_{n,2^n,\Psi_\alpha}$ with probability q. Then there is a PPT algorithm $\mathcal{A}$ that, given oracle access to $\mathcal{B}$ and a basis $\mathbf{B}$ for lattice $\mathcal{L}$, solves any BDD_γ instance $(\mathcal{L}, \mathbf{x})$ where*

$$\mathrm{dist}(\mathcal{L}^*, \mathbf{x}) \leq \frac{\alpha\gamma}{\max_{i\in[n]}\{\|\mathbf{b}_i\|\}} \cdot \left(\frac{1}{\pi} \ln\left(2n\left(1+\frac{1}{\varepsilon}\right)\right)\right)^{-\frac{1}{2}}$$

with probability $q(1+\delta)^{-3} - 6\varepsilon$ for some $\varepsilon \in (0, \frac{1}{24})$ and constant $\delta > \frac{3}{4}$.

In particular, using Conjecture 1, we assume that $q = 2^{-n^2/\kappa \log n}$. Then setting $\delta = 1$ and $\varepsilon = \frac{1}{96}q$, we obtain

$$q(1+\delta)^{-3} - 6\varepsilon = \frac{1}{8}q - \frac{1}{16}q = 2^{-4}q = 2^{-\frac{n^2}{\kappa \log n}-4} = 2^{-O(\frac{n^2}{\log n})}.$$

So informally, this theorem says that if there is no efficient algorithm that solves BDD_γ with probability $2^{-O(n^2/\log n)}$, then there is no efficient algorithm for $\mathrm{LWE}_{n,2^n,\Psi_\alpha}$ that succeeds with probability $2^{-O(n^2/\log n)}$.

The proof consists of two parts and uses techniques inspired by Regev's original reduction in [Reg09]. First we give a one-shot reduction from BDD to a generalised LWE problem in Sect. 3.1. Then we adapt Regev's reduction from this generalised LWE problem to search-LWEwith exponential modulus in Sect. 3.2, using the multiplicative, rather than additive, difference in distributions. Lastly, we reduce LWE with exponential modulus to binary LWE with polynomial modulus in Sect. 3.3.

3.1 BDD to Generalised LWE

Consider the following generalised version of LWE, as introduced by Regev in [Reg09].

Definition 18 (Generalised LWE). *The* Generalised Learning with Errors problem*, denoted by $\mathrm{LWE}_{n,p,\mathcal{D}}$, is defined as: given a polynomial number of samples from the distribution $A_{\mathbf{s},\phi}$ with modulus p, where ϕ belongs to the family of distributions $\mathcal{D}$, recover the secret $\mathbf{s} \in \mathbb{Z}_p^n$.*

Note that in the definition above, any algorithm for the problem may know $\mathcal{D}$, but is not given the specific distribution $\phi \in \mathcal{D}$. Furthermore in any instance of the problem, the input samples all come from the same distribution ϕ. For our proof, we are interested in the family of distributions

$$\Psi_{\leq\alpha} := \{\Psi_\beta \mid 0 < \beta \leq \alpha\}.$$

To obtain the desired bound on the success probability, we would like to minimise the number of calls to the oracle for our target problem. In the chain of reductions $\mathsf{BDD}_\gamma \rightarrow \mathsf{BDD}_{\gamma,p} \rightarrow \mathrm{LWE}_{n,p,\Psi_{\leq\alpha}}$ in [Reg09], a total of n calls is made to the algorithm for $\mathsf{BDD}_{\gamma,p}$, each of which calls the oracle for $\mathrm{LWE}_{n,p,\Psi_{\leq\alpha}}$

once. If we insist that the modulus is $p = 2^n$, then we can simplify our analysis by allowing the BDD_γ algorithm to call the $\mathsf{BDD}_{\gamma,p}$ oracle exactly once, so that the total number of calls to the LWE oracle is exactly one.

In our one-shot reduction from BDD_γ to $\mathsf{BDD}_{\gamma,p}$, we call the $\mathsf{BDD}_{\gamma,p}$ oracle on the given BDD_γ instance to obtain a coefficient vector that ideally corresponds to the closest lattice vector to the given target $\mathbf{v}$. We then shift the target vector $\mathbf{v}$ by this closest lattice vector and scale it down by p. Then we run Babai's nearest plane algorithm from [Bab86] on this shifted and scaled vector to find the closest lattice vector to $\mathbf{v}$ within the specified distance. Intuitively, blowing up the modulus to be exponential in the dimension n makes it easy for Babai's nearest plane algorithm to find the exact lattice point we are interested in. Now we formalise this idea.

Lemma 7. ($\mathsf{BDD} \to$ Modulo-BDD) *Let $n \in \mathbb{Z}_+, p = 2^n$, and $\gamma \in (0, \frac{1}{2})$. Suppose there is a polynomial time algorithm $\mathcal{B}$ that solves $\mathsf{BDD}_{\gamma,p}$ with success probability q. Then there exists a polynomial time algorithm $\mathcal{A}$ with oracle access to $\mathcal{B}$ that solves BDD_γ with success probability q.*

Proof. Let $(\mathbf{B}, \mathbf{v})$ be the given instance of BDD_γ. By definition, this defines a lattice $\mathcal{L} = \mathcal{L}(\mathbf{B}) \subset \mathbb{R}^n$ which satisfies $\mathrm{dist}(\mathbf{v}, \mathcal{L}) < \gamma \cdot \lambda_1(\mathcal{L})$. Consider the following algorithm $\mathcal{A}$:

Algorithm 1: BDD to Modulo-BDD Reduction

Input: BDD_γ instance $(\mathbf{B}, \mathbf{v})$.
Output: Lattice vector $\mathbf{x} \in \mathcal{L}$.
Run $\mathcal{B}$ on $(\mathbf{B}, \mathbf{v})$ to get a vector $\overline{\mathbf{z}} \in \mathbb{Z}_p^n$.
Compute $\mathbf{v}' := \frac{1}{p}(\mathbf{v} - \mathbf{B}\overline{\mathbf{z}})$.
Run Babai's Algorithm on $(\mathbf{B}, \mathbf{v}')$ to get a vector $\mathbf{Bz}' \in \mathcal{L}$.
Output the vector $\mathbf{B}(p\mathbf{z}' + \overline{\mathbf{z}})$.

Since the oracle $\mathcal{B}$ and Babai's Nearest Plane algorithm both run in time polynomial in the dimension of the lattice n, this algorithm clearly runs in polynomial time.

Now we prove correctness and show that if $\mathcal{B}$ answers correctly, then $\mathcal{A}$ outputs a correct answer. If oracle $\mathcal{B}$ answers correctly, it outputs $\overline{\mathbf{z}} = \mathbf{z} \pmod p$ for some coefficient vector $\mathbf{z} = \mathbf{B}^{-1}\mathbf{x} \in \mathcal{L}$, where $\mathbf{x} \in \mathcal{L}$ is a closest lattice vector to $\mathbf{v}$. This means that

$$\|\mathbf{v} - \mathbf{x}\| = \|\mathbf{v} - \mathbf{Bz}\| < \gamma \cdot \lambda_1(\mathcal{L}).$$

The output of $\mathcal{A}$ is correct if and only if $\|\mathbf{v} - \mathbf{B}(p\mathbf{z}' + \overline{\mathbf{z}})\| < \gamma \cdot \lambda_1(\mathcal{L})$. Since $\|\mathbf{v} - \mathbf{Bz}\| < \gamma \cdot \lambda_1(\mathcal{L})$, it is enough to show that $\mathbf{z} = p\mathbf{z}' + \overline{\mathbf{z}}$. Babai's nearest plane algorithm from [Bab86] guarantees that the output $\mathbf{Bz}'$ satisfies

$$\|\mathbf{v}' - \mathbf{Bz}'\| \leq 2^n \cdot \mathrm{dist}(\mathcal{L}, \mathbf{v}').$$

Observe that since $\mathbf{z} = \mathbf{B}^{-1}\mathbf{x}$ and $\mathbf{x} = \mathbf{Bz} \in \mathcal{L}$ is a lattice vector, $\mathbf{z} \in \mathbb{Z}^n$ must be an integer coefficient vector. By definition, $\overline{\mathbf{z}} = \mathbf{z} \pmod p \in \mathbb{Z}_p^n$ is

also a coefficient vector. Combining these two facts and observing that their coordinates can only differ by a multiple of p, we obtain that $\frac{1}{p}(\mathbf{z}-\overline{\mathbf{z}}) \in \mathbb{Z}^n$ is a coefficient vector. Hence $\mathbf{B}\frac{1}{p}(\mathbf{z}-\overline{\mathbf{z}}) \in \mathcal{L}$. By definition, $\mathrm{dist}(\mathcal{L}, \mathbf{v}') \leq \|\mathbf{v}'-\mathbf{u}\|$ for any lattice vector $\mathbf{u} \in \mathcal{L}$. This yields the bound

$$\|\mathbf{v}'-\mathbf{B}\mathbf{z}'\| \leq 2^n \cdot \mathrm{dist}(\mathcal{L}, \mathbf{v}') \leq 2^n \left\|\mathbf{v}' - \mathbf{B}\frac{1}{p}(\mathbf{z}-\overline{\mathbf{z}})\right\| = 2^n \cdot \frac{1}{p}\|\mathbf{v}-\mathbf{B}\mathbf{z}\| < \gamma \cdot \lambda_1(\mathcal{L}),$$

where the last inequality above follows from the assumption that $p = 2^n$. Thus, the unique closest vector to $\mathbf{v}'$ is in fact $\mathbf{B}\frac{1}{p}(\mathbf{z}-\overline{\mathbf{z}})$. Therefore, $\mathbf{z}' = \frac{1}{p}(\mathbf{z}-\overline{\mathbf{z}})$, so we obtain $\mathbf{z} = p\mathbf{z}' + \overline{\mathbf{z}}$ as desired. Since the oracle $\mathcal{B}$ is correct with probability q and the algorithm $\mathcal{A}$ always answers correctly in this case, the overall success probability of algorithm $\mathcal{A}$ is at least q. □

Note that in order for the reduction above to work, we need the modulus to be exponential $p = 2^n$. We will reduce this exponential modulus to a polynomial one in Sect. 3.3. Before we proceed to the second reduction, we introduce the following intermediate results.

Lemma 8. *(Claim 3.8 from [Reg09]) Let $\mathcal{L}$ be a rank-n lattice, $\mathbf{c} \in \mathbb{R}^n$, and $\varepsilon > 0$. For any $r \geq \eta_\varepsilon(\mathcal{L})$,*

$$\rho_r(\mathcal{L}+\mathbf{c}) \in (r^n \det(\mathcal{L}^*)(1-\varepsilon), r^n \det(\mathcal{L}^*)(1+\varepsilon)).$$

This bounds the Gaussian function of any lattice coset by the determinant of the dual lattice. The following lemma bounds the statistical distance between two relevant distributions.

Lemma 9. *(Corollary 3.10 from [Reg09]) Let $\mathcal{L}$ be a rank-n lattice, $\mathbf{w}, \mathbf{v} \in \mathbb{R}^n$ be vectors, and $r, s \in \mathbb{R}_+$. Define $t := \sqrt{(r\|\mathbf{w}\|)^2 + s^2}$. Suppose that for some $\varepsilon \in (0, \frac{1}{2})$, we have $\eta_\varepsilon(L) \leq 1/\sqrt{\frac{1}{r^2} + \left(\frac{1}{s}\|\mathbf{w}\|\right)^2}$. Define the random variable $X := \langle \mathbf{w}, \mathbf{v}\rangle + e$, where the distribution of $\mathbf{v}$ is $\mathbf{v} \sim D_{L+\mathbf{u},r}$ and $e \sim N\left(0, \frac{s^2}{2\pi}\right)$, and let Φ denote the distribution of X modulo 1. Also let $Z \sim N\left(0, \frac{t^2}{2\pi}\right)$. Then $\Delta(X, Z) \leq 4\varepsilon$, and hence $\Delta\left(\Phi, \Psi_t\right) \leq 4\varepsilon$.*

Now we implement the second reduction. Our algorithm requires additional data, namely samples from a discrete Gaussian. We generate these samples using a subroutine from [BLP+13] as a black-box. In [BLP+13], the authors give an efficient algorithm that, for any lattice and sufficiently large width, outputs a sample from a discrete Gaussian distribution. Formally, they prove the following.

Lemma 10. *(Theorem 2.3 from [BLP+13], adapted) There exists a PPT algorithm* DGS *that, given a basis $\mathbf{B}$ of an n-dimensional lattice $\mathcal{L} = \mathcal{L}(\mathbf{B})$, a vector $\mathbf{c} \in \mathbb{R}^n$, and a parameter*

$$r \geq \max_{i \in [n]}\{\|\tilde{\mathbf{b}}_i\|\} \cdot \sqrt{\frac{\ln(2n+4)}{\pi}},$$

outputs a sample with distribution $D_{\mathcal{L}+\mathbf{c},r}$.

Lemma 11. (Modulo-BDD $\rightarrow$ Generalised-LWE) *Let $\varepsilon = \varepsilon(n) \in (0, \frac{1}{24})$, $q = q(n)$, $\alpha = \alpha(n) \in (0,1)$, $p = p(n) \in \mathbb{Z}_+$, $\gamma \in (0, \frac{1}{2})$, and $k \in \mathbb{Z}_+$ be a constant. Suppose there is a polynomial time algorithm $\mathcal{B}$ that, given n^k samples from $A_{s,\Psi_{\leq\alpha}}$, solves $\mathrm{LWE}_{n,p,\Psi_{\leq\alpha}}$ with success probability q. Then there is a PPT algorithm $\mathcal{A}$ with oracle access to $\mathcal{B}$ that, given $(\mathbf{B}^*, \mathbf{x})$ corresponding to a lattice $\mathcal{L}^* = \mathcal{L}(\mathbf{B}^*)$ such that $\mathrm{dist}(\mathcal{L}^*, \mathbf{x}) \leq \frac{\alpha p\sqrt{\gamma}}{r}$ for some*

$$r > \frac{p}{\sqrt{\gamma}} \cdot \max_{i\in[n]}\{\|\mathbf{b}_i\|\} \cdot \sqrt{\frac{1}{\pi}\ln\left(2n\left(1+\frac{1}{\varepsilon}\right)\right)},$$

solves $\mathsf{BDD}_{\gamma,p}$ with success probability $q - 6\varepsilon$.

Proof. Let $(\mathbf{B}^*, \mathbf{x})$ be the given $\mathsf{BDD}_{\gamma,p}$ instance. By definition, this defines a lattice $\mathcal{L}^* = \mathcal{L}(\mathbf{B}^*) = (\mathcal{L}(\mathbf{B}))^*$ which is the dual lattice of $\mathcal{L} = \mathcal{L}(\mathbf{B})$. By Lemma 6, the parameter r is bounded by

$$r \geq \frac{p}{\sqrt{\gamma}\lambda_1(\mathcal{L}^*)} \cdot \sqrt{\frac{1}{\pi}\ln\left(2n\left(1+\frac{1}{\varepsilon}\right)\right)} \geq \frac{p}{\sqrt{\gamma}\lambda_1(\mathcal{L}^*)} \cdot \sqrt{\frac{1}{\pi}\ln\left(\frac{1}{\varepsilon}\right)} \geq \frac{\alpha p}{\sqrt{\gamma}\lambda_1(\mathcal{L}^*)}.$$

The last inequality here follows from the upper bound on ϵ and the assumption that $\alpha < 1$. Then for the given parameters, the distance between the given vector and lattice is bounded by

$$\mathrm{dist}(\mathcal{L}^*, \mathbf{x}) \leq \frac{\alpha p\sqrt{\gamma}}{r} \leq \frac{\alpha p\sqrt{\gamma} \cdot \sqrt{\gamma}\lambda_1(\mathcal{L}^*)}{\alpha p} \leq \gamma \cdot \lambda_1(\mathcal{L}^*).$$

Thus, $(\mathbf{B}^*, \mathbf{x})$ is a valid instance of $\mathsf{BDD}_{\gamma,p}$.

First we define a subroutine to efficiently sample from a discrete Gaussian distribution. Note that since $\varepsilon \in (0, \frac{1}{24})$ is a constant, we have $\frac{1}{\varepsilon} \geq \frac{2}{n}$, so the bound on r satisfies

$$r \geq \frac{p}{\sqrt{\gamma}} \cdot \max_{i\in[n]}\{\|\mathbf{b}_i\|\} \cdot \sqrt{\frac{1}{\pi}\ln\left(2n\left(1+\frac{1}{\varepsilon}\right)\right)} \geq \max_{i\in[n]}\{\|\mathbf{b}_i\|\} \cdot \sqrt{\frac{1}{\pi}\ln(2n+4)}.$$

This enables us to run the DGS algorithm from Lemma 10 on this r, the lattice $\mathcal{L}$, and vector $\mathbf{c} = 0$ to generate samples with distribution $D_{\mathcal{L},r}$.

The idea for algorithm $\mathcal{A}$ is to use $\mathbf{x}$ to generate a polynomial number of samples from a distribution Φ that is a good approximation of $A_{\mathbf{s},\Psi_\beta}$, for $\mathbf{s} = (\mathbf{B}^*)^{-1}\kappa_{\mathcal{L}^*}(\mathbf{x}) \bmod p$ and some $\beta \leq \alpha$. Recall that $\kappa_{\mathcal{L}^*}(\mathbf{x})$ denotes the unique closest vector in the lattice $\mathcal{L}^*$ to $\mathbf{x}$. We call the oracle $\mathcal{B}$ on these generated

samples to obtain the secret vector $\mathbf{s}$ with probability close to q. We formally define algorithm $\mathcal{A}$ as follows.

Algorithm 2: Modulo-BDD to Generalised-LWE Reduction

Input: $(\mathbf{B}^*, \mathbf{x})$ such that $\text{dist}(\mathcal{L}^*, \mathbf{x}) \leq \frac{\alpha p\sqrt{\gamma}}{r}$.
Output: $\mathbf{s} \in \mathbb{Z}_p^n$.
for $i \in \{1, ..., n^k\}$ **do**
  Run the DGS sampler to obtain a vector $\mathbf{v} \leftarrow D_{\mathcal{L},r}$.
  Compute $\mathbf{a} := \mathbf{B}^{-1}\mathbf{v} \bmod p$.
  Sample some noise $e \leftarrow N\left(0, \frac{\alpha^2\gamma}{2\pi}\right)$.
  Define $b := \frac{1}{p}\langle \mathbf{x}, \mathbf{v} \rangle + e \bmod 1$.
  Define sample $X_i := (\mathbf{a}, b)$.
end
Run $\mathcal{B}$ on $X_1, ..., X_{n^k} \sim \Phi$ to get a vector $\mathbf{s} \in \mathbb{Z}_p^n$.
Output $\mathbf{s}$.

Since this sampling process is efficient and repeated a polynomial number of times, the overall algorithm is efficient.

Now we prove the correctness of algorithm $\mathcal{A}$. We claim that if $\mathcal{B}$ succeeds, $\mathcal{A}$ generates a good approximation of samples from $A_{\mathbf{s},\Psi_\beta}$. Specifically, we show that the statistical distance between the distributions Φ and $A_{\mathbf{s},\Psi_\beta}$ is ε' for some $\beta \leq \alpha$. Given a polynomial number of samples from $A_{\mathbf{s},\Psi_\beta}$, the oracle $\mathcal{B}$ is guaranteed to find $\mathbf{s}$ with probability q. If the oracle succeeds, its output is $\mathbf{s} = (\mathbf{B}^*)^{-1}\kappa_{\mathcal{L}^*}(\mathbf{x}) \bmod p$, which is precisely the coefficient vector of the closest lattice vector $\mathbf{x} \in \mathcal{L}^*$ modulo p. Hence it is a solution for the given $\mathsf{BDD}_{\gamma,p}$ instance. Since the samples input to $\mathcal{B}$ are from an approximate distribution Φ that is ε' away in statistical distance from the true distribution $A_{\mathbf{s},\Psi_\beta}$, then by $(*)$ the success probability suffers a loss of ε'. Hence, the algorithm $\mathcal{A}$ will succeed with probability $q - \varepsilon'$.

We prove our claim by analysing the distributions of $\mathbf{a}$ and b for any generated sample X_i. First we show that the distribution of $\mathbf{a} \in \mathbb{Z}_p^n$ is close to uniform. Let $\mathcal{Y}$ denote the distribution of $\mathbf{a}$ produced in the algorithm. Fix $\mathbf{a} \in \mathbb{Z}_p^n$. Then the probability that $\mathcal{Y}$ takes the value $\mathbf{a}$ is

$$\Pr[\mathcal{Y} = \mathbf{a}] = \Pr_{\mathbf{v} \leftarrow D_{\mathcal{L},r}}[\mathbf{v} = \mathbf{Ba} \bmod p] = \frac{\rho_r(p\,\mathcal{L} + \mathbf{Ba})}{\rho_r(\mathcal{L})} = \frac{\rho_{\frac{r}{p}}\left(\mathcal{L} + \frac{1}{p}\mathbf{Ba}\right)}{\rho_r(\mathcal{L})},$$

by definition of the discrete Gaussian. By Lemma 5, we have $r > p\sqrt{2} \cdot \eta_\varepsilon(\mathcal{L})$. Then since $\eta_\varepsilon(\mathcal{L}) < r$, Lemma 8 implies

$$\frac{\rho_{\frac{r}{p}}\left(\mathcal{L} + \frac{1}{p}\mathbf{Ba}\right)}{\rho_r(\mathcal{L})} \in \frac{\frac{r^n}{p^n}\det(\mathcal{L}^*)(1 \pm \varepsilon)}{r^n \det(\mathcal{L}^*)(1 \pm \varepsilon)} = \frac{1}{p^n}\left(1 - \frac{2\varepsilon}{1+\varepsilon}, 1 + \frac{2\varepsilon}{1-\varepsilon}\right).$$

Then the statistical distance between $\mathcal{Y}$ and the uniform distribution $\mathcal{U}$ over $\mathbb{Z}_p^n$ is bounded by

$$\begin{aligned}\Delta(\mathcal{Y},\mathcal{U}) &= \frac{1}{2}\sum_{\mathbf{a}\in\mathbb{Z}_p^n} |\Pr[\mathcal{Y}=\mathbf{a}] - \Pr[\mathcal{U}=\mathbf{a}]| \\ &\le \frac{1}{2}\Bigg(\rho p^n\left(\frac{1}{p^n}\left(1+\frac{2\varepsilon}{1-\varepsilon}\right)-\frac{1}{p^n}\right) \\ &\quad + (1-\rho)p^n\left(\frac{1}{p^n}\left(1-\frac{2\varepsilon}{1+\varepsilon}\right)-\frac{1}{p^n}\right)\Bigg) \\ &\le \max_{\rho\in[0,1]}\left\{\rho\frac{\varepsilon}{1-\varepsilon}+(1-\rho)\frac{\varepsilon}{1+\varepsilon}\right\} \\ &\le \frac{\varepsilon}{1-\varepsilon}.\end{aligned}$$

Here $\rho \in [0,1]$ is the fraction of values in $\mathbb{Z}_p^n$ for which $\Pr[\mathcal{Y}] > \Pr[\mathcal{U}]$. Since $\varepsilon \in (0, \frac{1}{24})$, we have $\Delta(\mathcal{Y},\mathcal{U}) \le 2\varepsilon$.

Now we show that the distribution of the second component b of the sample X_i is close to the corresponding LWE distribution. We condition on $\mathbf{a}$ and consider the marginal distribution of b. Define $\mathbf{x}' := \mathbf{x} - \kappa_{\mathcal{L}^*}(\mathbf{x})$. By construction, we have $\|\mathbf{x}'\| \le \mathrm{dist}(\mathcal{L}^*, \mathbf{x}) \le \frac{\alpha p\sqrt{\gamma}}{r}$. Then we can write

$$\tfrac{1}{p}\langle \mathbf{x}, \mathbf{v}\rangle + e = \tfrac{1}{p}\langle \kappa_{\mathcal{L}^*}(\mathbf{x}), \mathbf{v}\rangle + \tfrac{1}{p}\langle \mathbf{x}', \mathbf{v}\rangle + e. \qquad (**)$$

Observe that since $(\mathbf{B}^*)^{\mathrm{T}} = \mathbf{B}^{-1}$, we can write

$$\begin{aligned}\langle \kappa_{\mathcal{L}^*}(\mathbf{x}), \mathbf{v}\rangle &= \kappa_{\mathcal{L}^*}(\mathbf{x})^{\mathrm{T}}\mathbf{B}\mathbf{B}^{-1}\mathbf{v} \\ &= \kappa_{\mathcal{L}^*}(\mathbf{x})^{\mathrm{T}}\big((\mathbf{B}^*)^{-1}\big)^{\mathrm{T}}\mathbf{B}^{-1}\mathbf{v} \\ &= \big((\mathbf{B}^*)^{-1}\kappa_{\mathcal{L}^*}(\mathbf{x})\big)^{\mathrm{T}}(\mathbf{B}^{-1}\mathbf{v}) \\ &= \big\langle (\mathbf{B}^*)^{-1}\kappa_{\mathcal{L}^*}(\mathbf{x}), \mathbf{B}^{-1}\mathbf{v}\big\rangle.\end{aligned}$$

By construction, we have

$$\langle \kappa_{\mathcal{L}^*}(\mathbf{x}), \mathbf{v}\rangle = \big\langle (\mathbf{B}^*)^{-1}\kappa_{\mathcal{L}^*}(\mathbf{x}), \mathbf{B}^{-1}\mathbf{v}\big\rangle \equiv \langle \mathbf{s}, \mathbf{a}\rangle \bmod p,$$

so the first term in (**) satisfies

$$\tfrac{1}{p}\langle \kappa_{\mathcal{L}^*}(\mathbf{x}), \mathbf{v}\rangle \equiv \tfrac{1}{p}\langle \mathbf{a}, \mathbf{s}\rangle \bmod 1.$$

It remains to consider the second term in the expression (**). Note that conditioned on $\mathbf{a}$ and since p is fixed, the distribution of $\mathbf{v}$ is the same as the distribution $D_{p\,\mathcal{L}+\mathbf{Ba},r}$. Let $\mathcal{Z}$ denote the distribution of $\frac{1}{p}\langle \mathbf{x}', \mathbf{v}\rangle + e \bmod 1$ for $\mathbf{v}$ sampled from $D_{p\,\mathcal{L}+\mathbf{Ba},r}$. By construction, e is sampled according to $N\left(0, \frac{\alpha^2\gamma}{2\pi}\right)$.

Since $\left\|\frac{1}{p}\mathbf{x}'\right\| \leq \frac{\alpha\sqrt{\gamma}}{r}$, we obtain

$$\frac{1}{\sqrt{\frac{1}{r^2} + \left(\frac{1}{\alpha\sqrt{\gamma}}\left\|\frac{1}{p}\mathbf{x}'\right\|\right)^2}} \geq \frac{1}{\sqrt{\frac{1}{r^2}+\frac{1}{r^2}}} = \frac{r}{\sqrt{2}} > r\sqrt{\gamma} \geq p \cdot \eta_\varepsilon(\mathcal{L}) = \eta_\varepsilon(p\,\mathcal{L}).$$

Here the last equality follows from rewriting the scaled smoothing parameter (see Subsect. 2.4 for the proof). Then by Lemma 9, $\Delta(\mathcal{Z}, \Psi_\beta) \leq 4\varepsilon$, for

$$\beta := \sqrt{\left(r\left\|\frac{1}{p}\mathbf{x}'\right\|\right)^2 + (\alpha\sqrt{\gamma})^2} \leq \sqrt{\alpha^2\gamma + \alpha^2\gamma} = \alpha\sqrt{2\gamma} < \alpha.$$

Therefore, by the triangle inequality, the statistical distance between Φ and $A_{\mathbf{s},\Psi_\beta}$ is $\varepsilon' = 2\varepsilon + 4\varepsilon = 6\varepsilon$ for some $\beta \leq \alpha$, as claimed. □

An immediate corollary of this one-shot reduction is the following bound on the success probability of any polynomial-time algorithm for BDD.

Corollary 1. (BDD → Generalised-LWE) *Suppose there exists an polynomial-time algorithm $\mathcal{A}$ that solves* $\mathrm{LWE}_{n,p,\phi}$*, where ϕ is an unknown distribution from the family $\Psi_{\leq\alpha}$, with success probability $q = q(n)$. Then there is a polynomial-time algorithm $\mathcal{B}$ that, given oracle access to $\mathcal{A}$, solves* BDD_γ *for gap $\gamma \in (0, \frac{1}{2})$ with probability $q - 6\varepsilon$ for some sufficiently small $\varepsilon = \varepsilon(n) \in (0, \frac{1}{24})$.*

Note that the additive loss in success probability can be written as a multiplicative factor:

$$q - \varepsilon' = q\left(1 - \frac{\varepsilon'}{q}\right) = q\left(1 - \frac{6\varepsilon}{q}\right).$$

This probability only makes sense for $0 < q - 6\varepsilon < 1$, which holds if $\epsilon \in \left(\frac{q-1}{6}, \frac{q}{6}\right)$. To obtain a small loss, say $q - \varepsilon' = \frac{q}{2}$, we would need $\epsilon = \frac{q}{12}$. For our application, we are interested in the regime where $q = 2^{-O(n^2/\log n)}$, so taking an appropriate ϵ such as this, we can obtain a constant multiplicative loss in success probability.

3.2 Generalised LWE to Standard LWE

In this section, we give a reduction from generalised LWE to LWE by adapting Lemma 3.7 from [Reg09] to work with multiplicative, rather than additive, loss in success probability. In Regev's reduction, we iteratively choose some Gaussian noise from a discrete interval to obtain some optimal noise that guarantees an overwhelming success probability. Since we are concerned with polynomial-time adversaries and the success probability itself, unlike the original reduction, we only sample noise from the interval exactly once. Because we are limited to a single Gaussian noise sample, choosing the interval and parameters requires considerable care.

Lemma 12. (Generalised-LWE → Search-LWE) *Let $\alpha \in \mathbb{R}_+$, $p = p(n) \in \mathbb{Z}_+$, and $\varepsilon > \frac{3}{4}$ be a constant parameter. Suppose there is an efficient algorithm $\mathcal{B}$ that solves* $\mathrm{LWE}_{n,p,\Psi_\alpha}$ *with success probability q. Then there is a PPT algorithm $\mathcal{A}$ that, given oracle access to $\mathcal{B}$, solves* $\mathrm{LWE}_{n,p,\Psi_{\leq\alpha}}$ *with success probability at least $\frac{q}{(1+\varepsilon)^3}$.*

Proof. Suppose that $\mathcal{A}$ is given n^k samples for some constant $k \in \mathbb{Z}_+$, distributed according to $A_{\mathbf{s},\Psi_\beta}$, for some $\beta \leq \alpha$. For notational convenience, let $\delta := (1+\varepsilon)^2 - 1 = \varepsilon^2 + 2\varepsilon$ and define

$$Z := \left\{0, \delta\alpha^2, 2 \cdot \delta\alpha^2, \ldots, \lfloor\delta\rfloor\delta\alpha^2\right\}$$

to be the set of integer multiples of $\delta\alpha^2$ between 0 and α^2. Consider the following algorithm $\mathcal{A}$:

Algorithm 3: Generalised-LWE to Search-LWE Reduction

Input: Samples $X_1, ..., X_{n^k} \sim A_{\mathbf{s},\Psi_\beta}$.
Output: Secret vector $\mathbf{s} \in \mathbb{Z}_p^n$.
Sample $\gamma \leftarrow Z$ uniformly at random.
for $i \in \{1, ..., n^k\}$ **do**
  Denote $(\mathbf{a}, b) := X_i$.
  Sample some noise $e \leftarrow \Psi_{\sqrt{\gamma}}$.
  Define $Y_i := (\mathbf{a}, b + e)$.
end
Run $\mathcal{B}$ on the generated samples $Y_1, ..., Y_{n^k}$ to get a vector $\mathbf{s}' \in \mathbb{Z}_p^n$.
Output $\mathbf{s}'$.

Since sampling and transforming n^k samples is efficient, and the oracle $\mathcal{B}$ is called once, $\mathcal{A}$ is efficient.

Now we prove correctness of $\mathcal{A}$. The algorithm is given samples of the form $X_i = (\mathbf{a}, b) = (\mathbf{a}, \langle\mathbf{a}, \mathbf{s}\rangle + e)$, where e has distribution Ψ_β for some unknown $\beta \leq \alpha$. The algorithm knows the value of α, so it attempts to add noise from $\Psi_{\sqrt{\gamma}}$ in such a way as to obtain samples with noise distribution Ψ_α. In this way, it generates samples of the form $Y_i = (\mathbf{a}, b + e) = (\mathbf{a}, \langle\mathbf{a}, \mathbf{s}\rangle + e' + e)$ where the noise $e' + e$ has distribution Ψ_σ for $\sigma := \sqrt{\beta^2 + \gamma}$. The error between the noise distribution Ψ_σ generated by $\mathcal{A}$ and the target distribution Ψ_α is determined by the given error parameter ε. Let γ' be the smallest element of Z satisfying $\gamma' \geq \alpha^2 - \beta^2$. Then by construction of Z, we have $\gamma' \leq \alpha^2 - \beta^2 + \delta\alpha^2 = (\delta + 1)\alpha^2 - \beta^2$. By definition, since $\varepsilon > \frac{3}{4}$, we have $\delta > 1$. Together with the fact that $0 < \beta \leq \alpha$, this implies that $0 < \alpha^2 - \beta^2 < (\delta + 1)\alpha^2 - \beta^2 < \lfloor\delta\rfloor\delta\alpha^2$. Hence, there exists such an element γ' in Z. There are $|Z| = \frac{\lfloor\delta\rfloor\delta\alpha^2}{\delta\alpha^2} \leq \delta$ elements in Z, so the element γ sampled by the algorithm is $\gamma = \gamma'$ with probability at least $\frac{1}{\delta}$. Let $\sigma' := \sqrt{\beta^2 + \gamma'}$ denote the noise distribution parameter for this special element γ'. Consider the ratio of the probability generating functions corresponding to

Ψ_α and $\Psi_{\sigma'}$. Using Lemma 2 and the bounds on γ', this is given by

$$\frac{g_\alpha(x)}{g_{\sigma'}(x)} \leq \frac{\sigma'}{\alpha} = \frac{\sqrt{\beta^2+\gamma'}}{\alpha}$$
$$\leq \frac{\sqrt{\beta^2+(\delta+1)\alpha^2-\beta^2}}{\alpha} = \frac{\alpha\sqrt{1+\delta}}{\alpha} = \sqrt{(1+\varepsilon)^2} = 1+\varepsilon.$$

Then by Lemma 3, applying any function to this ratio of probability distribution functions cannot increase the ratio. This implies that the success probability of $\mathcal{A}$ for noise distribution Ψ_α and the success probability of $\mathcal{A}$ for noise distribution $\Psi_{\sigma'}$ have the ratio

$$\frac{\Pr[\mathcal{A} \text{ succeeds for } \Psi_\alpha]}{\Pr[\mathcal{A} \text{ succeeds for } \Psi_{\sigma'}]} = \frac{q}{\Pr[\mathcal{A} \text{ succeeds for } \gamma=\gamma']} \leq 1+\varepsilon.$$

Hence, for this choice $\gamma = \gamma'$, we know that $\mathcal{A}$ successfully outputs $\mathbf{s}' = \mathbf{s}$ with probability at least $\frac{q}{1+\varepsilon}$. Therefore the overall success probability of $\mathcal{A}$ is at least

$$\Pr[\mathcal{A} \text{ succeeds for } \gamma=\gamma'] \cdot \Pr[\gamma=\gamma'] \geq q \cdot \frac{1}{1+\varepsilon} \cdot \frac{1}{\delta} \geq \frac{q}{(1+\varepsilon)^3}.$$

□

Assuming the success probability of solving $\mathrm{LWE}_{n,p,\Psi_\alpha}$ is $q = p^{-n/\kappa \log n}$ for modulus $p = 2^n$, and setting the error parameter to be $\varepsilon = 1$, we obtain the following corollary.

Corollary 2. (Generalised-LWE → Search-LWE) *Suppose there is no efficient algorithm $\mathcal{A}$ for $\mathrm{LWE}_{n,2^n,\Psi_{\leq\alpha}}$ with success probability $2^{-n^2/\kappa \log n - 3}$ for some constant $c > 0$. Then there is no efficient algorithm $\mathcal{B}$ for $\mathrm{LWE}_{n,2^n,\Psi_\alpha}$ with success probability $2^{-n^2/\kappa \log n}$.*

3.3 Reducing the Modulus for Search-LWE

In [BLP+13], Brakerski et al. study the trade-off between the modulus and dimension of decision-LWE instances. In particular, they give a reduction from decision-LWE to decision-LWE that reduces the modulus arbitrarily while preserving the dimension and incurring only a small loss in advantage. Their reduction can also be viewed as a search to search reduction that says the following.

Theorem 4. *(Theorem 4.1. from [BLP+13], rephrased) Let $n \in \mathbb{Z}_+$ and $\alpha = \alpha(n) \in (0,1)$ such that $\frac{1}{\alpha}$ is bounded by a polynomial in n. Then for some prime $p = p(n)$ such that both p and $\frac{p}{\alpha}$ are $n^{\Theta(1)}$, there is a polynomial-time, one-shot reduction from $\mathrm{LWE}_{n,2^n,\Psi_\alpha}$ to $\mathrm{binLWE}_{n^2,p,\Psi_\alpha}$ that preserves the success probability.*

Using the trivial reduction from binLWE to LWE for the same dimension, modulus, and noise distribution, this result allows us to reduce the modulus from exponential to polynomial in n for LWE. For completeness, we include this simple reduction below.

Lemma 13. ($\text{binLWE}_{n,p,\phi} \rightarrow \text{LWE}_{n,p,\phi}$) *Suppose there exists an efficient algorithm $\mathcal{B}$ that solves $\text{LWE}_{n,p,\phi}$ with success probability q. Then there is a PPT algorithm $\mathcal{A}$ that, given oracle access to $\mathcal{B}$, solves $\text{binLWE}_{n,p,\phi}$ with success probability q.*

Proof. Let $A_{\mathbf{s},\phi}$ be the input distribution for the given $\text{binLWE}_{n,p,\phi}$ samples, where $\mathbf{s} \in \{0,1\}^n$ is a binary secret vector. Consider the following algorithm $\mathcal{A}$:

Algorithm 4: binLWE to LWE Reduction

Input: Samples $X_1, ..., X_{n^k} \sim A_{\mathbf{s},\phi}$.
Output: Secret vector $\mathbf{s} \in \mathbb{Z}_p^n$.
Sample a vector $\mathbf{r} \leftarrow \mathbb{Z}_p^n$ uniformly at random.
for $i \in \{1, ..., n^k\}$ **do**
  Denote $(\mathbf{a}, b) := X_i$.
  Define $Y_i := (\mathbf{a}, b + \langle \mathbf{a}, \mathbf{r} \rangle)$.
end
Run $\mathcal{B}$ on the generated samples $Y_1, ..., Y_{n^k}$ to get a vector $\mathbf{s}' \in \mathbb{Z}_p^n$.
Output $\mathbf{s}' - \mathbf{r}$.

This algorithm transforms a polynomial number of samples and the efficient oracle $\mathcal{B}$ is called once, so $\mathcal{A}$ is efficient. Now we prove correctness. Observe that each sample X_i has $b = \langle \mathbf{a}, \mathbf{s} \rangle + e$ for some noise e with distribution ϕ, so the transformed samples have the form

$$Y_i = (\mathbf{a}, b + \langle \mathbf{a}, \mathbf{r} \rangle) = (\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle + \langle \mathbf{a}, \mathbf{r} \rangle + e) = (\mathbf{a}, \langle \mathbf{a}, \mathbf{s} + \mathbf{r} \rangle + e).$$

The oracle $\mathcal{B}$ succeeds in recovering the secret vector $\mathbf{s}' = \mathbf{s} + \mathbf{r}$ with probability q, so with the same probability $\mathcal{A}$ outputs the secret binary vector $\mathbf{s} = \mathbf{s}' - \mathbf{r}$. □

4 Search-LWE to Decision-LWE

In this section, we show how to solve search-LWE given an oracle for decision-LWE, under the condition that the oracle correctly responds YES far more often than it incorrectly responds YES. The formal statement of our result is below. Several proofs in this section have been omitted for lack of space, so we refer the reader to the full version (available online) for details.

Theorem 5. (Search-LWE $\rightarrow$ Decision-LWE) *Let $n, p, k \in \mathbb{Z}_+$ be such that $p > 10$ is polynomial in n and k is a constant. Suppose that there exists an efficient algorithm $\mathcal{B}$ for decision-$\text{LWE}_{n,p,\phi}$ that,*

- *given n^k LWE samples from $A_{\mathbf{s},\phi}$, outputs YES with probability γ,*
- *given n^k random samples from $\mathbb{Z}_p^n \times \mathbb{T}$, outputs YES with probability δ,*

where $\gamma > 5p^2\delta$. Then there is an algorithm $\mathcal{A}$ for search-$\text{LWE}_{n,p,\phi}$ with oracle access to $\mathcal{B}$ that, given n^k samples, runs in expected polynomial time and

- *outputs a correct answer with probability $\frac{1}{5p^3}\gamma$ and*

– *outputs* $\perp$ *with probability* $1 - \frac{1}{5p^3}\gamma$.

Note that here we do not make any assumptions on how large γ and δ, so they need not be negligibly small. We prove this by making the following key observation: If solving search-LWE is hard, then it is hard to determine the secret vector $\mathbf{s}$ from a given polynomial number of LWE samples of the form $(\mathbf{a}, \mathbf{b} = \frac{1}{p}\langle \mathbf{a}, \mathbf{s}\rangle + e)$. Intuitively, this means that the function $f_\phi = (\mathbf{A}, \frac{1}{p}\mathbf{As} + \mathbf{e})$ defined by these samples is hard to invert, so it can be viewed as a one-way function. In their seminal work, Goldreich and Levin show how to construct a hard-core predicate from any one-way function [GL89]. This tells us that if we can find the inner product of $\mathbf{s}$ and a given vector $\mathbf{r}$, then we can recover the secret vector $\mathbf{s}$.

Inspired by this connection, we define an intermediate problem we call the *Goldreich-Levin Learning with Errors* (GL-LWE) problem. We reduce search-LWE with polynomial modulus to GL-LWE, then reduce this problem to standard decision-LWE under a reasonable condition.

4.1 Search-LWE to GL-LWE

In [GL89], Goldreich and Levin showed that for any one-way function f, the function $b(\mathbf{x}, \mathbf{z}) := \langle \mathbf{x}, \mathbf{z}\rangle \bmod 2$ is a hard-core predicate for the function $g(\mathbf{x}, \mathbf{z}) := (f(\mathbf{x}), \mathbf{z})$. Levin later improved this result in [Lev12] and showed that the success probability of finding the hard-core predicate is determined by the success probability of inverting the one-way function f. For the formal statement and full proof of Levin's result, we refer the reader to the full version of our paper.

We generalise Levin's result from modulus 2 to modulus $p > 10$ using the natural generalisation of a hard-core predicate for $\mathbb{Z}_p$, under a certain condition. We refer the reader to the full version for our proof of this generalisation.

Lemma 14. *Let* $n, p \in \mathbb{Z}_+$ *such that* $p > 10$ *is polynomial in* n *and let* $f : \mathbb{Z}_p^n \to \mathbb{Z}_p^n$ *be an injective one-way function. Suppose there is an efficient algorithm* $\mathcal{B}$ *that, given* $\mathbf{y} = f(\mathbf{x})$ *for some* $\mathbf{x} \in \mathbb{Z}_p^n$ *and random* $\mathbf{r} \in \mathbb{Z}_p^n$*, guesses* $\langle \mathbf{x}, \mathbf{r}\rangle \bmod p$

– *correctly with probability* $\alpha\beta$,
– *incorrectly with probability* $\alpha(1 - \beta)$, *and*
– *outputs* $\perp$ *with probability* $1 - \alpha$,

where $\beta > 1 - \frac{1}{5p}$*. The probability is taken over the randomness of* $\mathbf{r}$*, and the randomness of the algorithm. Then there is an algorithm* $\mathcal{A}$ *that runs in expected polynomial time, that given oracle access to* $\mathcal{B}$ *and* $\mathbf{y} = f(\mathbf{x})$ *for some* $\mathbf{x} \in \mathbb{Z}_p^n$*, finds* $\mathbf{x}$ *correctly with probability* $\frac{1}{5p^2}\alpha\beta$ *and outputs* $\perp$ *with probability* $1 - \frac{1}{5p^2}\alpha\beta$.

Now we apply this result to our study of the hardness of LWE. First we define an intermediate worst-case problem inspired by the Goldreich-Levin theorem.

Definition 19 (Goldreich-Levin LWE). *The Goldreich-Levin Learning with Errors (GL-LWE) problem, denoted by* GL-LWE$_{n,p,\phi}$ *is defined as: given a polynomial number of samples from the distribution* $A_{\mathbf{s},\phi}$*, where* $\mathbf{s} \in \mathbb{Z}_p^n$ *is some fixed secret, and a uniformly random vector* $\mathbf{r} \in \mathbb{Z}_p^n$*, find* $\langle \mathbf{s}, \mathbf{r} \rangle \bmod p$*.*

We use this to reduce (average-case) search-LWE to (average-case) decision-LWE. First note that search-LWE trivially reduces to worst-case search-LWE: given LWE samples for a uniformly random secret $\mathbf{s}$, the reduction algorithm simply runs its oracle for worst-case search-LWE on the given samples and succeeds in recovering $\mathbf{s}$ with the same probability. Now we interpret Lemma 14 as a reduction from worst-case search-LWE to GL-LWE.

Corollary 3. (Search-LWE → GL-LWE) *Let* $n, p, k \in \mathbb{Z}_+$ *such that* $p > 10$ *is polynomial in* n*. Suppose that there is an efficient algorithm* $\mathcal{B}$ *for* GL-LWE$_{n,p,\phi}$ *that, given* n^k *samples from* $A_{\mathbf{s},\phi}$ *for some fixed secret* $\mathbf{s} \in \mathbb{Z}_p^n$*, and a uniformly random vector* $\mathbf{r} \in \mathbb{Z}_p^n$*, outputs a guess for* $\langle \mathbf{s}, \mathbf{r} \rangle \bmod p$

- *correctly with probability* $\alpha^*\beta^*$*,*
- *incorrectly with probability* $\alpha^*(1-\beta^*)$*, and*
- *outputs* $\perp$ *with probability* $1-\alpha^*$*,*

where $\beta^* > 1 - \frac{1}{5p}$*. The probability is taken over the randomness of* $\mathbf{s}, \mathbf{r}$*, and the randomness of the algorithm. Then there is an algorithm* $\mathcal{A}$ *for search-*LWE$_{n,p,\phi}$ *that runs in expected polynomial time and, given* n^k *samples from* $A_{\mathbf{s},\phi}$ *for some fixed secret* $\mathbf{s} \in \mathbb{Z}_p^n$*,*

- *correctly outputs* $\mathbf{s}$ *with probability* $\frac{1}{5p^2}\alpha\beta$ *and*
- *outputs* $\perp$ *with probability* $1 - \frac{1}{5p^2}\alpha\beta$*.*

Proof. Consider the function $f_\phi : \mathbb{Z}_p^n \to \mathbb{Z}_p^n$ given by $f_\phi(\mathbf{s}) := (\mathbf{A}, \frac{1}{p}\mathbf{A}\mathbf{s} + \mathbf{e})$, where the rows of $\mathbf{A}$ are uniformly random vectors sampled from $\mathbb{Z}_p^n$ and $\mathbf{e}$ is sampled according to distribution ϕ. This can be used as an injective function, because with high probability there is a unique $\mathbf{s}$ that satisfies the system of equations determined by any given output $(\mathbf{A}, \mathbf{b}) = (\mathbf{A}, \frac{1}{p}\mathbf{A}\mathbf{s} + \mathbf{e})$. Applying Lemma 14 gives us the desired result. □

4.2 GL-LWE to Decision-LWE

Finally, we finish our chain of reductions by reducing our worst-case intermediate problem to decision-LWE. In the following we show that if there is a (γ, δ)-solver (with $\gamma \gg \delta$) for decision-LWE, then there is an algorithm for GL-LWE that Corollary 3 can be instantiated with to complete the reduction from search-LWE to decision-LWE.

Lemma 15. (GL-LWE → Decision-LWE) *Let* $k \in \mathbb{Z}_+$ *be a constant. Suppose that there exists an efficient algorithm* $\mathcal{B}$ *for decision-*LWE$_{n,p,\phi}$ *that,*

- *given* n^k LWE *samples from* $A_{\mathbf{s},\phi}$ *for a uniformly random secret* $\mathbf{s}$*, outputs YES with probability* γ*,*

– *given* n^k *random samples from* $\mathbb{Z}_p^n \times \mathbb{T}$*, outputs YES with probability* δ*.*

Then there is an algorithm $\mathcal{A}$ *that, given oracle access to* $\mathcal{B}$ *and an instance of* GL-LWE$_{n,p,\phi}$ *consisting of* n^k *samples from* $A_{\mathbf{s},\phi}$ *for a fixed secret* $\mathbf{s}$*, outputs*

– *a correct answer with probability* $\frac{1}{p}\gamma$*,*
– *a wrong answer with probability* $\frac{p-1}{p}\delta$*, and*
– $\perp$ *with probability* $1 - \frac{1}{p}\gamma - \frac{p-1}{p}\delta$*.*

For the proof of Lemma 15, we refer the reader to the full version. Now we combine Corollary 3 and Lemma 15 to obtain our second main result.

Corollary 4. (Search-LWE $\rightarrow$ Decision-LWE) *Let* $n, p, k \in \mathbb{Z}_+$ *be such that* $p > 10$ *is polynomial in* n *and* k *is a constant. Suppose that there exists an efficient algorithm* $\mathcal{B}$ *for decision-*LWE$_{n,p,\phi}$ *that,*

– *given* n^k LWE *samples from* $A_{\mathbf{s},\phi}$*, outputs YES with probability* γ*,*
– *given* n^k *random samples from* $\mathbb{Z}_p^n \times \mathbb{T}$*, outputs YES with probability* δ*,*

where $\gamma > 5p^2\delta$*. Then there is an algorithm* $\mathcal{A}$ *for search-*LWE$_{n,p,\phi}$ *with oracle access to* $\mathcal{B}$ *that, given* n^k *samples, runs in expected polynomial time and*

– *outputs a correct answer with probability* $\frac{1}{5p^3}\gamma$ *and*
– *outputs* $\perp$ *with probability* $1 - \frac{1}{5p^3}\gamma$*.*

Proof. Set $\alpha := \frac{\gamma+(p-1)\delta}{p}$ and $\beta := \frac{\gamma}{\gamma+(p-1)\delta}$. Then we have $\alpha\beta = \frac{1}{p}\gamma$ and $\alpha(1-\beta) = \frac{p-1}{p}\delta$. By the assumption that $\gamma > 5p^2\delta$, and since $p > 10$, we obtain

$$\beta = \frac{\gamma}{\gamma + (p-1)\delta} > 1 - \frac{p-1}{5p^2 + p - 1} > 1 - \frac{1}{5p}.$$

Consider the following algorithm $\mathcal{A}$: Given an instance of search-LWE, first run the algorithm from Lemma 15 to solve the corresponding instance of GL-LWE. Then run the algorithm from Corollary 3 to solve the corresponding GL-LWE instance. Finally, run the trivial algorithm to solve the given average-case search-LWE instance. By Corollary 3, for these values of α and β, this algorithm $\mathcal{A}$ outputs a correct answer for the given instance of search-LWE with probability

$$\frac{1}{5p^2}\alpha\beta = \frac{1}{5p^3}\gamma,$$

and outputs $\perp$ with the remaining probability. □

5 Conclusions and Future Directions

In this paper, we offer a new perspective on the computational complexity of lattice problems by revisiting the notion of characterizing the hardness of a

computational problem in terms of the maximum success probability achievable by any probabilistic polynomial-time algorithm.

We show how characterising hardness in such a way enables us to obtain a much tighter reduction from the worst-case BDD problem for lattices to the average-case search-LWE problem, as well as a tight reduction from search-LWE to decision-LWE. (See Sect. 1.3 for precise statements.)

We believe that our work should motivate quantifying the hardness of computational problems − especially those relevant to cryptography − using a similar metric. We emphasize that such reductions will be very sensitive to the number of calls made to the oracle, since the success probability will decrease exponentially with the number of oracle calls. For the reductions in this work, our main challenge was to ensure that our reductions make a single call to the oracle, even if that meant the reduction succeeds with a relatively small probability.

Acknowledgments. The authors would like to thank Chris Peikert, Vinod Vaikuntanathan, Leo Ducas, and Daniele Micciancio for helpful conversations during the course of this work. This project was supported partially by NSF awards CCF-2236931 and CCF-2107345 and NRF award NRF-NRFI09-0005.

References

[ABGSD21] Aggarwal, D., Bennett, H., Golovnev, A., Stephens-Davidowitz, N.: Fine-grained hardness of CVP(P) – Everything that we can prove (and nothing else) (2021)

[ADRS14] Aggarwal, D., Dadush, D., Regev, O., Stephens-Davidowitz, N.: Solving the shortest vector problem in 2^n time via discrete gaussian sampling. CoRR, abs/1412.7994 (2014)

[ADRS15] Aggarwal, D., Dadush, D., Regev, O., Stephens-Davidowitz, N.: Solving the shortest vector problem in 2^n time via discrete gaussian sampling (2015)

[Ajt96] Ajtai, M.: Generating hard instances of lattice problems. Electron. Colloquium Comput. Complex. TR96 (1996)

[ALNSD20] Aggarwal, D., Li, J., Nguyen, P.Q., Stephens-Davidowitz, N.: Slide reduction, revisited—filling the gaps in SVP approximation. In: Annual International Cryptology Conference, pp. 274–295. Springer (2020)

[ALS20] Aggarwal, D., Li, Z., Stephens-Davidowitz, N.: A $2^{n/2}$-Time Algorithm for $\sqrt{n}$-SVP and $\sqrt{n}$-Hermite SVP, and an Improved Time-Approximation Tradeoff for (H)SVP. arXiv e-prints (2020)

[AM11] Aggarwal, D., Maurer, U.: The leakage-resilience limit of a computational problem is equal to its unpredictability entropy. In: Advances in Cryptology–ASIACRYPT 2011: 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, 4-8 December 2011. Proceedings 17, pp. 686–701. Springer (2011)

[ASD18] Aggarwal, D., Stephens-Davidowitz, N.: (Gap/S)ETH Hardness of SVP. In: Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, pp. 228–238, New York, NY, USA (2018). Association for Computing Machinery

[Bab86] Babai, L.: On Lovász lattice reduction and the nearest lattice point problem. Combinatorica **6**, 1–13 (1986)

[BGS17] Bennett, H., Golovnev, A., Stephens-Davidowitz, N.: On the Quantitative Hardness of CVP. CoRR, abs/1704.03928 (2017)

[BKW00] Blum, A., Kalai, A., Wasserman, H.; Noise-tolerant learning, the parity problem, and the statistical query model. CoRR, cs.LG/0010022 (2000)

[BLP+13] Brakerski, Z., Langlois, A., Peikert, C., Regev, O., Stehlé, D.: Classical hardness of learning with errors. In: Proceedings of the Forty-Fifth Annual ACM Symposium on Theory of Computing, STOC 2013, pp. 575–584, New York, NY, USA (2013). Association for Computing Machinery

[Bri83] Brickell, E.F.: Solving low density knapsacks. In: Advances in Cryptology: Proceedings of CRYPTO 1983, pp. 25–37. Plenum (1983)

[BV14] Brakerski, Z., Vaikuntanathan, V.: Lattice-based FHE as secure as PKE. In: Proceedings of the 5th Conference on Innovations in Theoretical Computer Science, ITCS 2014, pp. 1–12, New York, NY, USA (2014). Association for Computing Machinery

[Cai98] Cai, J.-Y.: A relation of primal-dual lattices and the complexity of shortest lattice vector problem. Theoret. Comput. Sci. **207**(1), 105–116 (1998)

[Din02] Dinur, I.: Approximating $\mathrm{SVP}_{\mathrm{infinity}}$ to within almost-polynomial factors is NP-hard. Theor. Comput. Sci. **285**(1), 55–71 (2002)

[DKRS03] Dinur, I., Kindler, G., Raz, R., Safra, S.: Approximating CVP to within almost-polynomial factors is NP-Hard. Combinatorica **23**(2), 205–243 (2003)

[FT87] Frank, A., Tardos,É.: An application of simultaneous diophantine approximation in combinatorial optimization. Combinatorica, **7**(1), 49–65 (1987)

[Gen09] Gentry, C.: Fully Homomorphic Encryption Using Ideal Lattices. In: Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing, STOC 2009, pp. 169–178, New York, NY, USA (2009). Association for Computing Machinery

[GL89] Goldreich, O., Levin, L.A.: A hard-core predicate for all one-way functions. In: Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing, STOC 1989, pp. 25–32, New York, NY, USA (1989). Association for Computing Machinery

[GMSS99] Goldreich, O., Micciancio, D., Safra, S., Seifert, J.-P.: Approximating shortest lattice vectors is not harder than approximating closest lattice vectors. Inf. Process. Lett. **71**(2), 55–61 (1999)

[GN08] Gama, N., Nguyen, P.Q.: Finding short lattice vectors within Mordell's inequality. In: Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing, pp. 207–216 (2008)

[GPV08] Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing, STOC 2008, pp. 197–206, New York, NY, USA (2008). Association for Computing Machinery

[HR18] Haviv, I., Regev, O.: Tensor-based hardness of the shortest vector problem to within almost polynomial factors. CoRR, abs/1806.04087 (2018)

[Kan87] Kannan, R.: Minkowski's convex body theorem and integer programming. Math. Oper. Res. **12**(3), 415–440 (1987)

[Kho05] Khot, S.: Hardness of approximating the shortest vector problem in lattices. J. ACM **52**(5), 789–808 (2005)

[Lev12] Levin, L.A.: Randomness and non-determinism. CoRR, abs/1211.0071 (2012)

[LLL82] Lenstra, A., Lenstra, H., László, L.: Factoring polynomials with rational coefficients. Math. Ann. **261**, 12 (1982)

[LM09] Lyubashevsky, V., Micciancio, D.: On bounded distance decoding, unique shortest vectors, and the minimum distance problem. In: Shai Halevi, (ed.) Advances in Cryptology - CRYPTO 2009, pp. 577–594. Springer, Heidelberg (2009)

[LO85] Lagarias, J., Odlyzko, A.: Solving low-density subset sum problems. J. ACM **32**, 229–246 (1985)

[Mic12] Micciancio, D.: Inapproximability of the shortest vector problem: toward a deterministic reduction. Theory Comput. **8**(1), 487–512 (2012)

[MM11] Micciancio, D., Mol, P.: Pseudorandom knapsacks and the sample complexity of LWE search-to-decision reductions. Cryptology ePrint Archive, Paper 2011/521 (2011)

[MR04] Micciancio, D., Regev, O.: Worst-case to average-case reductions based on Gaussian measures. In: 45th Annual IEEE Symposium on Foundations of Computer Science, pp. 372–381 (2004)

[MR09] Micciancio, D., Regev, O.: Lattice-based Cryptography, pp. 147–191. Springer, Heidelberg (2009)

[MV13] Micciancio, D., Voulgaris, P.: A deterministic single exponential time algorithm for most lattice problems based on Voronoi cell computations. SIAM J. Comput. **42**(3), 1364–1391 (2013)

[MW18] Micciancio, D., Walter, M.: On the bit security of cryptographic primitives. In Jesper Buus Nielsen and Vincent Rijmen, (eds.) Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part I, vol. 10820 LNCS, pp. 3–28. Springer (2018)

[Pei09] Peikert, C.: Public-key cryptosystems from the worst-case shortest vector problem. In: Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing, pp. 333–342 (2009)

[PP10] Paturi, R., Pudlák, P.: On the Complexity of Circuit Satisfiability. In: Proceedings of the Forty-Second ACM Symposium on Theory of Computing, STOC 2010, pp. 241–250, New York, NY, USA (2010). Association for Computing Machinery

[Reg06] Regev, O.: Lattice-based cryptography. In Cynthia Dwork, (ed.) Advances in Cryptology - CRYPTO 2006, pp. 131–141. Springer, Heidelberg (2006)

[Reg09] Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. J. ACM **56**(6), 1–40 (2009)

[Sha85] Shamir, A.: Identity-based cryptosystems and signature schemes. In: George Robert Blakley and David Chaum, (eds.) Advances in Cryptology, pp. 47–53. Springer, Heidelberg, (1985)

[Vad12] Vadhan, S.P.: Pseudorandomness. Found. Trends Theor. Comput. Sci. **7**(1–3), 1–336 (2012)

[vEB81] van Emde Boas, P.: Another NP-complete problem and the complexity of computing short vectors in a lattice. Technical Report, Department of Mathmatics, University of Amsterdam (1981)

Proofs II

Batching Adaptively-Sound SNARGs for NP

Lalita Devadas[1(✉)], Brent Waters[2,3], and David J. Wu[2]

[1] MIT, Cambridge, MA, USA
lali@mit.edu
[2] UT Austin, Austin, TX, USA
[3] NTT Research, Sunnyvale, CA, USA

Abstract. A succinct non-interactive argument (SNARG) for NP allows a prover to convince a verifier that an NP statement x is true with a proof whose size is *sublinear* in the length of the traditional NP witness. Moreover, a SNARG is adaptively sound if the adversary can choose the statement it wants to prove after seeing the scheme parameters. Very recently, Waters and Wu (STOC 2024) showed how to construct adaptively-sound SNARGs for NP in the plain model from falsifiable assumptions (specifically, sub-exponentially secure indistinguishability obfuscation, sub-exponentially secure one-way functions, and polynomial hardness of discrete log).

We consider the *batch* setting where the prover wants to prove a collection of T statements $x_1, \ldots, x_T$ and its goal is to construct a proof whose size is sublinear in both the size of a single witness *and* the number of instances T. In this setting, existing constructions either require the size of the public parameters to scale linearly with T (and thus, can only support an a priori bounded number of instances), or only provide non-adaptive soundness, or have proof size that scales linearly with the size of a single NP witness. In this work, we give two approaches for batching adaptively-sound SNARGs for NP, and in particular, show that under the same set of assumptions as those underlying the Waters-Wu adaptively-sound SNARG, we can obtain an adaptively-sound SNARG for batch NP where the size of the proof is $\mathsf{poly}(\lambda)$ and the size of the CRS is $\mathsf{poly}(\lambda + |C|)$, where λ is a security parameter and $|C|$ is the size of the circuit that computes the associated NP relation.

Our first approach builds directly on top of the Waters-Wu construction and relies on indistinguishability obfuscation and a *homomorphic* re-randomizable one-way function. Our second approach shows how to combine ideas from the Waters-Wu SNARG with the chaining-based approach by Garg, Sheridan, Waters, and Wu (TCC 2022) to obtain a SNARG for batch NP.

1 Introduction

Succinct non-interactive arguments (SNARGs) for NP allow an efficient prover to convince a verifier that an NP statement x (with associated witness w) is true

E. Boyle and M. Mahmoody (Eds.): TCC 2024, LNCS 15365, pp. 339–370, 2025.
https://doi.org/10.1007/978-3-031-78017-2_12

with a proof whose size scales with $o(|x| + |w|)$. The main security requirement is computational soundness which says that a computationally-bounded prover should not be able to convince a verifier of a false statement. SNARGs were first constructed in the random oracle model [31,34]. Many works have subsequently shown how to construct SNARGs in the plain model assuming the prover and the verifier have access to a common reference string (CRS).

Until recently, SNARGs for NP in the CRS model have either relied on non-falsifiable cryptographic assumptions (c.f., [1,3–7,15,16,19,21,32] and the references therein) or satisfied the weaker notion of *non-adaptive* soundness [37], where soundness only holds against an adversary that declares its false statement before seeing the CRS. In contrast, the standard or "adaptive" notion of soundness allows the malicious prover to choose the statement *after* seeing the CRS. Very recently, several works gave the first adaptively-sound SNARGs for NP using indistinguishability obfuscation ($i\mathcal{O}$) and either a sub-exponentially-secure re-randomizable one-way function [40] or a sub-exponentially-secure lossy function [42].[1] Moreover, in the *designated-verifier* model where a secret key is needed to verify proofs, the work of [33] shows that the original Sahai-Waters scheme (based on $i\mathcal{O}$ and one-way functions) [37] is also adaptively sound. In conjunction with constructions of $i\mathcal{O}$ from falsifiable cryptographic assumptions [24,25], these works provide the first adaptively-sound SNARGs for NP from falsifiable assumptions.

Batch Arguments. Existing constructions of adaptively-sound SNARGs for NP focus on the single-statement setting where the prover constructs a proof for a single statement. In many settings (e.g., incrementally verifiable computation [38] or proof-carrying data [11]), a prover might have a batch of T (possibly correlated) statements $x_1, \ldots, x_T$ that it wants to prove to the verifier, and the goal is to construct a single short proof (whose size is sublinear in T *and* in the size of the associated NP relation) of all T statements. There are two main approaches to constructing batch arguments:

- **Using BARGs for NP:** Non-interactive batch arguments (BARGs) for NP [13,14,28,29] provide one possible approach. Namely, a BARG for NP allows a prover to prepare a proof on T statements with a proof whose size scales sublinearly (ideally, polylogarithmically) with the number of statements T. Moreover, many recent works have shown how to construct BARGs for NP from a broad range of cryptographic assumptions [12–14,17,23,26,27,29,36,39]. However, in these existing constructions, the size of the proof grows with the size of the circuit that decides a *single* statement, and the goal is to amortize the proof size across the number of statements. Allowing the proof size to grow with the size of the NP relation avoids black-box separations that pertain to SNARGs for NP [20]. In this work, we are

[1] A subsequent work [41] also shows how to construct an adaptively-sound SNARG using $i\mathcal{O}$ and sub-exponentially-secure one-way functions without any additional algebraic assumptions.

interested in batching SNARG proofs, where the size of the proof is sublinear in both the number of statements *and* size of the circuit computing the NP relation; such arguments are said to be *fully* succinct [18]. The previous work of [18] showed how to construct fully succinct BARGs for NP using $i\mathcal{O}$ and one-way functions, but the construction only achieved non-adaptive soundness.

– **Using SNARGs for NP:** Another approach to constructing a fully succinct SNARG for a batch language is to view the batch statement $(x_1, \ldots, x_T)$ as a *single* NP statement for a product language (i.e., the statement $(x_1, \ldots, x_T)$ is in the language if for each $i \in [T]$, there exists a valid witness w_i for x_i), and then use a SNARG for NP to prove the product language. This approach achieves adaptive soundness if we instantiate the underlying SNARG with an adaptively-sound SNARG for NP [40,42]. However, the size of the CRS in existing adaptively-sound SNARGs [40,42] grows polynomially with the size of the NP relation circuit. Thus, if we directly apply an existing adaptively-sound SNARG for NP to a batch language, the NP relation circuit would take all T statements as input, and the size of the CRS scales polynomially with T. This means the CRS is large and moreover, there is an *a priori* bound on the number of statements that can be batched. In this work, our goal is to support aggregating an *arbitrary* polynomial number of (adaptively-sound) proofs on NP statements.

Why Not Compose? If we settle for non-adaptive soundness, the work of [18] shows that we can construct a fully succinct SNARG for batch languages by composing a standard (somewhere-extractable) BARG for NP with a SNARG for NP. Namely, a proof on statements $(x_1, \ldots, x_T)$ is a BARG proof that there exists SNARG proofs $\pi_1, \ldots, \pi_T$ for the statements $x_1, \ldots, x_T$. In this case, the NP relation associated with the BARG is the SNARG verification circuit, which is small by construction. Moreover, if the BARG is *somewhere extractable* [14][2] and the SNARG is non-adaptively sound, then it is straightforward to show that the composed scheme satisfies non-adaptive soundness. While we can replace the underlying SNARG in this composition with an adaptively-sound construction, we are not able to prove adaptive soundness for the composition. The issue is that if we rely on somewhere extractability for the BARG, then the reduction needs to "know" the index of the false statement and program it into the CRS; this is not possible when the statements are adaptively chosen.

Alternatively, we could consider a reduction algorithm that guesses the index of the false statement. Since the index is computationally hidden from the malicious prover, the hope would be that a prover that consistently chooses statements $(x_1, \ldots, x_T)$ that evades the guess (i.e., where the index of the false state-

[2] A BARG is somewhere extractable if the CRS can be programmed on a (hidden) index $i \in [T]$. Then, given a valid BARG proof π on a batch of statements $(x_1, \ldots, x_T)$, there is an efficient extraction algorithm that recovers a witness w_i for x_i. The special index i is computationally hidden by the CRS. Somewhere extractable BARGs can be constructed from most number-theoretic assumptions [12–14,17,23,26,27,29,36,39].

ment is different from the guessed index) must be breaking index hiding of the somewhere extractable BARG. The problem is that checking whether the adversary successfully evaded the guess (and thus, broke index hiding) is not an efficient procedure (it requires deciding the underlying NP statement). We could handle this by complexity leveraging and relying on a super-polynomial time reduction that is able to decide the underlying NP relation. However, if we do so, then the size of the resulting BARG starts scaling with the size of the NP relation, and the resulting construction is no longer succinct.

This Work. In this work, we show how to construct adaptively-sound SNARGs for batch languages with almost no overhead compared to the single-statement setting. Specifically, we show how to leverage the adaptively-sound SNARG for NP from [40] to obtain an adaptively-sound SNARG for batch languages with only polylogarithmic additive overhead in the number of statements T. We summarize our instantiation in the following (informal) theorem:

Theorem 1 (Informal). *Let λ be a security parameter. Assuming (1) the polynomial hardness of computing discrete logs in a prime-order group, (2) the existence of a sub-exponentially-secure indistinguishability obfuscation scheme for Boolean circuits, and (3) the existence of a sub-exponentially-secure one-way function, there exists an adaptively-sound SNARG for batch NP with the following properties:*

- **Preprocessing SNARG:** *Let $C\colon \{0,1\}^n \times \{0,1\}^v \to \{0,1\}$ be the circuit that computes the NP relation (where n is the statement size and v is the witness size). The size of the common reference string for proving up to $T \leq 2^\lambda$ statements is $\mathsf{poly}(\lambda + |C|)$.*
- **Proof Size:** *A proof on a batch of $T \leq 2^\lambda$ statements $(x_1, \ldots, x_T)$ has size $\mathsf{poly}(\lambda)$.*

Additionally, the SNARG is perfect zero-knowledge.

The Gentry-Wichs Separation. The classic result of Gentry and Wichs [20] gives a barrier for constructing adaptively-sound SNARGs for NP from falsifiable assumptions where the running time of the reduction is insufficient to decide the underlying NP language. Consequently, existing constructions of adaptively-sound SNARGs for NP [33,40,42] all rely on complexity leveraging and super-polynomial-time security reductions. In these constructions, the cost of the complexity leveraging is incurred in the size of the CRS. In the setting of batch NP, the time it takes to decide a batch of T statements $(x_1, \ldots, x_T)$ is only a factor of T greater than the time it takes to decide a single statement. As such, obtaining an adaptively-sound SNARG for batch NP would only increase the running time of the reduction algorithm by a factor of T. In this case, the size of the CRS (or the proof) would only need to increase by a factor of $\log T$. In contrast, for a general NP relation where the statements and witnesses are a factor of T longer, the reduction may have to run in time that is greater by a factor 2^T to decide the larger language, which would lead to a CRS that is larger by a factor of $\mathsf{poly}(T)$ rather than $\mathsf{poly}(\log T)$.

1.1 Technical Overview

We begin by describing the Waters-Wu [40] adaptively-sound SNARG for NP based on indistinguishability obfuscation ($i\mathcal{O}$) and re-randomizable one-way functions. Throughout, we consider the language of Boolean circuit satisfiability, where the Boolean circuit $C\colon \{0,1\}^n \times \{0,1\}^v \to \{0,1\}$ is fixed ahead of time (i.e., as part of the CRS). A statement $x \in \{0,1\}^n$ is true if there exists a witness $w \in \{0,1\}^v$ such that $C(x,w) = 1$.

Building Blocks. In addition to $i\mathcal{O}$, the [40] construction requires a puncturable pseudorandom function (PRF) [8,9,30], and a re-randomizable one-way function. In a puncturable PRF $\mathsf{F}(k,\cdot)$, the holder of the secret key k can "puncture" the key at an input point x^* to create a punctured key $k^{(x^*)}$. The punctured key $k^{(x^*)}$ can be used to evaluate $\mathsf{F}(k,x)$ on all points $x \neq x^*$. However, the value $\mathsf{F}(k,x^*)$ at the punctured point remains pseudorandom even given the punctured key $k^{(x^*)}$. The final ingredient they require is a re-randomizable one-way function (OWF) f. This is a OWF equipped with a *statistical* re-randomization algorithm that takes as input a OWF challenge y_{base} and produces a fresh challenge y (sampled uniformly at random from the challenge space of the OWF). Moreover, given the re-randomization randomness together with a solution to the re-randomized statement, there is an efficient algorithm for recovering a solution to the original OWF challenge y_{base}. In other words, the re-randomization can be viewed as a (perfect) random self-reducibility property on the OWF.

The Waters-Wu Construction. In the Waters-Wu construction, the CRS consists of two obfuscated programs: (1) a "solution-generator" program GenSol used to construct proofs; and (2) a "challenge-generator" program $\mathsf{GenChall}$ used to verify proofs. The solution-generator GenSol has the circuit C (for the NP relation) together with three puncturable PRF keys $k_{\mathsf{sel}}, k_0, k_1$ hard-wired inside.

At a high level, in the Waters-Wu construction, the proof on a statement x is a pair $(b, \mathsf{F}(k_b, x))$ where $b \in \{0,1\}$. The solution-generator program takes as input a bit b, statement x, and witness w, and checks that $b \neq \mathsf{F}(k_{\mathsf{sel}}, x)$ and that $C(x,w) = 1$. If so, it outputs the solution $(b, \mathsf{F}(k_b, x))$. The challenge-generator program takes as input a bit b and statement x and outputs the challenge $y_b = f(\mathsf{F}(k_b, x))$.

The idea is that the solution-generator program only ever outputs *one* of the two possible solutions associated with each statement x. Moreover, which one it chooses is determined pseudorandomly by evaluating the selector PRF $\mathsf{F}(k_{\mathsf{sel}}, x)$. We will sometimes refer to the challenge y_b associated with $b = \mathsf{F}(k_{\mathsf{sel}}, x)$ as the "on-path" challenge for x and the challenge y_b associated with $b = 1 - \mathsf{F}(k_{\mathsf{sel}}, x)$ as the "off-path" challenge for x.

In the Waters-Wu construction, the prover program is constructed so it only provides solutions to the off-path challenge; the prover program never generates a solution to an on-path challenge. Then, in the proof of adaptive soundness, [40] show how to replace the on-path challenge statement for *every* statement with a re-randomized statement of a one-way function. The hope is that if the malicious

prover ever produces a proof for a false statement x that corresponds to the on-path challenge, then it successfully breaks the one-way function. Finally, the [40] analysis appeals to the fact that for a false statement x, the value of the selector PRF $\mathsf{F}(k_{\mathsf{sel}}, x)$ is computationally unpredictable to the adversary; as such, with probability $1/2$, the prover will provide a solution to the on-path statement, which completes the proof. We now give the formal description of the GenSol and $\mathsf{GenChall}$ programs:[3]

$\mathsf{GenSol}(b, x, w)$	$\mathsf{GenChall}(b, x)$
– If $C(x, w) = 0$, output $\perp$. – If $b = \mathsf{F}(k_{\mathsf{sel}}, x)$, output $\perp$. – Output $z = \mathsf{F}(k_b, x)$.	– Output $y = f(\mathsf{F}(k_b, x))$.

To construct a proof for a statement x and witness w, the prover simply runs the (obfuscated) GenSol program on input $(0, x, w)$ and input $(1, x, w)$. GenSol will output $\perp$ on one of these inputs, and an OWF preimage $z = \mathsf{F}(k_b, x)$ on the other. The proof $\pi = (b, z)$ consists of the bit b and the preimage z. To check the proof π, the verifier simply runs the (obfuscated) $\mathsf{GenChall}$ program on input (b, x). $\mathsf{GenChall}$ will output a OWF challenge $y = f(\mathsf{F}(k_b, x))$, and the verifier checks that $f(z) = y$.

We now sketch the proof of soundness from [40]. As mentioned above, the proof proceeds in a sequence of hybrid experiments. First, we argue that with probability $1/2$, the malicious prover will output an on-path solution as its proof; this is because for a false statement x, it is unable to predict the value of $\mathsf{F}(k_{\mathsf{sel}}, x)$. Next, we gradually replace the on-path challenge for every statement program with a re-randomized one-way function challenge. This way, a solution to *any* on-path challenge implies a solution to the original one-way function challenge. Since the GenSol program never outputs an on-path solution, this does not affect completeness. However, if the prover ever produces an on-path solution, then it successfully inverts the one-way function and adaptive soundness follows. We now sketch the sequence of hybrids:

- Hyb_0: This is the real adaptive soundness game. The challenger outputs 1 only if the adversary $\mathcal{A}$ produces an accepting proof $\pi = (b, z)$ for a false statement x: namely, $f(z) = y = \mathsf{GenChall}(b, x)$.
- Hyb_1: After the adversary $\mathcal{A}$ outputs its proof $\pi = (b, z)$, the challenger additionally checks that $b = \mathsf{F}(k_{\mathsf{sel}}, x)$, or in other words, that $\mathcal{A}$ output a solution to the on-path challenge. This can only reduce $\mathcal{A}$'s success probability by a factor of 2, since the value of $\mathsf{F}(k_{\mathsf{sel}}, x)$ is computationally hidden from

[3] Note that the original Waters-Wu construction did not require GenSol and $\mathsf{GenChall}$ to take the bit $b \in \{0, 1\}$ as input. Instead, GenSol computed $b = \mathsf{F}(k_{\mathsf{sel}}, x)$ and outputted $z = \mathsf{F}(k_b, x)$ while $\mathsf{GenChall}$ simply outputted $f(\mathsf{F}(k_0, x))$ and $f(\mathsf{F}(k_1, x))$. The adaptation here is equivalent to the original Waters-Wu construction and the updated syntax will be conducive when extending to batch arguments.

the adversary for every false statement x (by puncturing security). Formally, [40] show this by considering an exponential sequence of hybrids, one for each false statement x^*. In $\mathsf{Hyb}_1^{(x^*)}$, the challenger punctures k_{sel} at x^* and hard-wires the punctured key $k_{\mathsf{sel}}^{(x^*)}$ in GenSol instead of k_{sel}:

$\mathsf{GenSol}^{(x^*)}(b, x, w)$	$\mathsf{GenChall}(b, x)$
• If $C(x, w) = 0$, output $\perp$. • If $b = \mathsf{F}(k_{\mathsf{sel}}^{(x^*)}, x)$, output $\perp$. • Output $z = \mathsf{F}(k_b, x)$.	• Output $y = f(\mathsf{F}(k_b, x))$.

When x^* is a false statement, $\mathsf{GenSol}^{(x^*)}$ still computes the same functionality as GenSol: both immediately reject, since there does not exist a w such that $C(x^*, w) = 1$. Thus, $\mathsf{GenSol}^{(x^*)}$ does not need to evaluate $\mathsf{F}(k_{\mathsf{sel}}, x^*)$. Now, by puncturing security, the value of $\mathsf{F}(k_{\mathsf{sel}}, x^*)$ is pseudorandom even given $k_{\mathsf{sel}}^{(x^*)}$. Thus, if the adversary outputs a proof $\pi = (b, z)$ for x^*, with probability $1/2 - \mathsf{negl}(\lambda)$, it will be the case that $\mathsf{F}(k_{\mathsf{sel}}, x^*) = b$.

– Hyb_2: In this experiment, the challenger stops checking whether or not x is false; observe that this can only increase the adversary's success probability. In addition, the challenger samples a random OWF challenge $y_{\mathsf{base}} \leftarrow f(r)$ for uniform r along with a puncturable PRF key k_{rerand} that will be used to re-randomize y_{base}. The challenger now modifies $\mathsf{GenChall}$ to output a re-randomization of y_{base} on (b, x) whenever $b = \mathsf{F}(k_{\mathsf{sel}}, x)$. In other words, the on-path challenges are now replaced by a re-randomized instance of y_{base}. To argue that this is computationally indistinguishable from the previous hybrid, the [40] reduction again steps through an exponential number of hybrids, one for each statement x^*. Planting the re-randomized challenge is then an exercise in punctured programming [37]. The key observation is that the GenSol program never evaluates $\mathsf{F}(k_b, x^*)$ for $b = \mathsf{F}(k_{\mathsf{sel}}, x^*)$. We can then appeal to punctured pseudorandomness of $\mathsf{F}(k_b, x^*)$ to conclude that the challenge y_b is computationally indistinguishable from a fresh one-way function challenge, which is in turn statistically indistinguishable from a re-randomized instance.

In Hyb_2, algorithm $\mathcal{A}$ can only succeed if it provides a solution to a re-randomized one-way function instance. But this means that $\mathcal{A}$ also inverts the original one-way function challenge, which completes the proof of adaptive security. Observe that here, polynomial security of the one-way function already suffices. Importantly, this final step is the only step in the analysis that relies on one-wayness. Thus, the proof π remains succinct despite the use of an exponential number of hybrids in the previous steps. The exponential sequence of hybrids require blowing up the security parameters for the $i\mathcal{O}$ and puncturable PRF schemes, but this only affects the length of the CRS and *not* the proof.

Our First Approach: Batching SNARGs Using Homomorphic One-Way Functions. We now show how to extend the Waters-Wu scheme to the batch setting.

Recall that in this setting, the prover has a collection of T statements $x_1, \ldots, x_T$ and its goal is to prove that all T statements are true. If we directly modify the GenSol and $\mathsf{GenChall}$ programs above to take in all T statements, then the resulting CRS would have size that scales linearly with T, and moreover, the scheme would only support an a priori bounded number of statements. Our goal is to obtain a construction without this limitation.

As described above, the Waters-Wu construction uses a re-randomizable one-way function. In [40], they show two instantiations of the re-randomizable one-way function: the first is based on the hardness of discrete log while the second is based on factoring. In this work, we will consider the construction based on discrete log. To recall, let $\mathbb{G}$ be a group of prime-order p and let g be a generator of $\mathbb{G}$. The one-way function then $f\colon \mathbb{Z}_p \to \mathbb{G}$ is then defined to be the mapping $z \mapsto g^z$. Our first observation is that this one-way function is homomorphic:

$$f(z_1 + z_2) = g^{z_1+z_2} = g^{z_1} \cdot g^{z_2} = f(z_1) \cdot f(z_2).$$

In the context of the Waters-Wu SNARG, the values z would correspond to the preimages in the proof π. Suppose now that we have T proofs $(b_1, z_1), \ldots, (b_T, z_T)$ on T different statements $x_1, \ldots, x_T$. Then a natural approach to obtain a batch proof on all T statements by computing $z = \sum_{i\in[T]} z_i \in \mathbb{Z}_p$. Then,

$$f(z) = f\left(\sum_{i\in[T]} z_i\right) = \prod_{i\in[T]} f(z_i) = \prod_{i\in[T]} y_{i,b_i},$$

where $y_{i,b_i} = \mathsf{GenChall}(b_i, x_i)$ is the challenge bit associated with statement i. Now, if the verifier knew the bits $b_1, \ldots, b_T$, it can compute $y_{i,b_i} = \mathsf{GenChall}(b_i, x_i)$ and then $y = \prod_{i\in[T]} y_{i,b_i} \in \mathsf{G}$. Then, the verification algorithm would simply boil down to checking that $y = f(z)$. In this case, the prover just needs to provide the *aggregated* preimage z rather than the individual preimages $(z_1, \ldots, z_T)$.

The problem, of course, is that the verifier does *not* know the individual bits $b_i \in \{0, 1\}$. While the prover can certainly include the bits b_i for each statement as part of the proof, this means the size of the proof is now $T + \mathsf{poly}(\lambda)$, which no longer meets our succinctness requirement. Note that if $T = O(\log \lambda)$, the verifier can try all the possible values for $b_1, \ldots, b_T$, but this approach does not work for general T. We solve this problem by *increasing the alphabet size*. Namely, instead having two challenges, suppose instead we had $T + 1$ challenges (i.e., the selection PRF $\mathsf{F}(k_{\mathsf{sel}}, \cdot)$ now outputs an element of the set $[T + 1]$). In this setting, for each statement x, there is still a single "on-path" challenge (the index determined by $\mathsf{F}(k_{\mathsf{sel}}, x) \in [T+1]$) for which the GenSol program will not provide a preimage and T off-path challenges for which the GenSol program will provide preimages (if also given a valid witness for x). This means that for any batch of T statements, there will exist some index $j \in [T+1]$ for which $j \neq \mathsf{F}(k_{\mathsf{sel}}, x_i)$ for all $i \in [T]$. Since the same index j can now be *shared* across all T statements, the prover only needs to communicate the single index (of length $O(\log T)$) as part of its proof. Concretely, we modify the programs in the CRS as follows:

$\mathsf{GenSol}(j, x, w)$	$\mathsf{GenChall}(j, x)$
– If $C(x, w) = 0$, output $\perp$. – If $j = \mathsf{F}(k_{\mathsf{sel}}, x)$, output $\perp$. – Output $z = \mathsf{F}(k_j, x)$.	– Output $y = f(\mathsf{F}(k_j, x))$.

Concretely, our scheme now operates as follows:

- **Proof generation:** To construct a proof on $x_1, \ldots, x_T$ (given witnesses $w_1, \ldots, w_T$), the prover first finds an index $j \in [T+1]$ where

$$z_i = \mathsf{GenSol}(j, x_i, w_i) \neq \perp$$

  for all $i \in [T]$. Then it computes the aggregated proof $z = \sum_{i \in [T]} z_i$ and outputs the proof $\pi = (j, z)$.
- **Proof Verification:** To verify the proof, the verifier computes the challenge

$$y_i = \mathsf{GenChall}(j, x_i)$$

  for each $i \in [T]$ and then computes the aggregated challenge $y = \prod_{i \in [T]} y_i$. Finally, the verifier checks that $g^z = y$.

As written, the GenSol and $\mathsf{GenChall}$ programs would require us to hard-wire $T+1$ puncturable PRF keys $k_1, \ldots, k_{T+1}$ into the GenSol and $\mathsf{GenChall}$ programs (to derive the $T+1$ possible challenges). Consequently, the size of the CRS now grows with T, which is no better than directly applying [40] to the batch language. To get around this, we derive the keys k_j for $j \in [T+1]$ from another (puncturable) PRF. The modified programs are defined as follows:

$\mathsf{GenSol}(j, x, w)$	$\mathsf{GenChall}(j, x)$
– If $C(x, w) = 0$, output $\perp$. – If $j = \mathsf{F}(k_{\mathsf{sel}}, x)$, output $\perp$. – Compute $k_j \leftarrow \mathsf{F}(k, j)$. – Output $z = \mathsf{F}(k_j, x)$.	– Compute $k_j \leftarrow \mathsf{F}(k, j)$. – Output $y = f(\mathsf{F}(k_j, x))$.

In the soundness proof, when we modify $\mathsf{GenChall}$ to output a re-randomized one-way function challenge as the on-path challenge for each statement x^* (i.e., the transition from Hyb_1 to Hyb_2 in the above description), we proceed in two steps. First, we puncture the key-derivation PRF key k at the on-path index $j^* = \mathsf{F}(k_{\mathsf{sel}}, x^*)$ and hard-wire the punctured keys $k_{j^*}^{(x^*)}$ and $k^{(j^*)}$ into the GenSol and $\mathsf{GenChall}$ programs. As in the Waters-Wu analysis, the GenSol program never computes $\mathsf{F}(k_{j^*}, x^*)$, so its value remains pseudorandom. We give the formal description and analysis of this scheme in Sect. 4.

Observe that the size of the programs in the CRS is $\mathsf{poly}(\lambda + |C| + \log T)$. Thus, setting $T = 2^{\lambda}$ allows us to support any a priori unbounded polynomial number of statements. This gives the first adaptively-sound SNARG for batch NP (that supports an unbounded number of statements) with full succinctness from standard falsifiable assumptions.

Our Second Approach: Batching by Chaining Using Re-randomizable PRGs. Thus far, we have demonstrated how to extend the Waters-Wu SNARG to support batching by relying on the homomorphic structure of the one-way function. In this work, we also give a second approach to support batching that does not assume any homomorphic properties on the output SNARG. Instead, our construction relies on a re-randomizable pseudorandom generator (PRG), which we define more precisely below.

Here, we follow a similar template as the general aggregation approach from [18]. The work of [18] constructs a non-adaptively-sound SNARG for batch NP by adapting the non-adaptively-sound SNARG for NP by Sahai and Waters [37]. Specifically, they describe a "chaining" approach where the prover program (in the CRS) takes as input a hash digest dig of the statements $(x_1, \ldots, x_T)$, a proof π_{i-1} on the first $i-1$ statements, the next statement x_i and associated witness w_i, together with an opening of x_i with respect to the digest dig. The prover checks that π_{i-1} is a valid proof on the digest dig, that dig opens to x_i at position i, and that w_i is a valid witness for x_i. If all of these properties hold, then the program outputs a proof for the first i statements (with respect to the digest dig).

To prove non-adaptive soundness, the idea in [18] is to first identify the index $i \in [T]$ of a *false* statement, and use punctured programming to argue that there does not *exist* any accepting proofs on the first i statements. This step relies on the fact that there are no witnesses for the false statement x_i. Then, they show that if there does not exist an accepting proof for index i, there also does not exist an accepting proof for index $i+1$. This proceeds until the final hybrid where they argue that there does not exist any proof for index T, at which point non-adaptive soundness holds.

Unlike the single-statement setting, in the chaining approach, it is no longer sufficient to argue that an accepting proof of a false statement is computationally hard to find. This is because the obfuscated prover program (i.e., the analog of GenSol) is first checking the proof on the first $i-1$ statements when deciding whether to generate a proof for the first i statements or not. If there exists a valid proof on the first $i-1$ statements, then this program does not output $\perp$ on inputs with index i (e.g., consider the setting where statement x_i is true). As a result, we cannot argue that there does not exist a proof on the first i statements. In contrast, if we can argue that there are no accepting proofs on the first $i-1$ statements, then we can leverage $i\mathcal{O}$ security to argue that there are also no accepting proofs on the first i statements (since the obfuscated prover program never accepts a proof on the first $i-1$ statements, it would never output a proof for the first i statements). In the Waters-Wu approach, they showed that if an adversary could construct a proof of a false statement, then the adversary can

also invert the one-way function. Notably, this is a *computational* property, and the previous analysis can only rule out an adversary finding an accepting proof efficiently. Consequently, this is insufficient to implement the chaining approach from [18] as proofs of *false* statements do exist (but are hard to find). The work of [18] leverages an (expanding) pseudorandom generator to check the proofs instead of using a one-way function precisely to move to a hybrid where proofs on false statements no longer exist.

In Sect. 6, we show how to use a similar chaining strategy together with the Waters-Wu approach to obtain an adaptively-sound SNARG for batch NP. For the reasons outlined above, our approach requires a way to rule out the *existence* of proofs on false statements. For this reason, we rely on the stronger notion of a re-randomizable PRG instead of a re-randomizable OWF. In a re-randomizable PRG $\mathsf{G}\colon \{0,1\}^\lambda \to \{0,1\}^t$, there is an algorithm that takes a string $y_{\mathsf{base}} \in \{0,1\}^t$ and re-randomizes it to a new string $y \in \{0,1\}^t$ with the following properties:

- If y_{base} is the in the image of the PRG (i.e., $y_{\mathsf{base}} = \mathsf{PRG}(s)$ for some $s \in \{0,1\}^\lambda$), then the re-randomized value y is distributed according to $\mathsf{G}(s)$ for a fresh seed $s \xleftarrow{R} \{0,1\}^\lambda$.
- If y_{base} is not in the image of the PRG, then the re-randomized value y is distributed according to a random value $y \xleftarrow{R} \{0,1\}^t \setminus \{\mathsf{PRG}(s) : s \in \{0,1\}^\lambda\}$.

We can construct a re-randomizable PRF from the decisional Diffie-Hellman (DDH) assumption. In particular, we work over a group $\mathbb{G}$ of prime order p and generator g, and define the public parameters to be (g, h) where $h \xleftarrow{R} \mathbb{G}$. Then, we define the generator $\mathsf{G}\colon \mathbb{Z}_p \to \mathbb{G} \times \mathbb{G}$ as the mapping $x \mapsto (g^x, h^x)$. Pseudorandomness follows directly from the DDH assumption, and the re-randomization follows via the DDH random self-reduction.

In our construction, we replace the GenSol program with a proof aggregation program $\mathsf{AggProof}$, which only outputs a proof for a digest dig and index i if it is given a valid proof on dig and index $i-1$, a valid statement-witness pair (x_i, w_i), and dig opens to statement x_i at index i. The $\mathsf{GenChall}$ program is replaced by a proof verification program Verify, which will only accept a proof (j, z) for the digest dig and index i if $\mathsf{G}(z) = \mathsf{G}(\mathsf{F}(k_j, (\mathsf{dig}, i)))$. Importantly, we have replaced the re-randomizable one-way function f with the re-randomizable PRG. If we instantiate this template with the adaptively-sound construction with $T + 1$ challenges, the CRS will consist of obfuscations of the following programs:

$\mathsf{AggProof}(j, \mathsf{dig}, i, x, w, \sigma, z_{i-1})$	$\mathsf{Verify}(j, \mathsf{dig}, i, z_i)$
– If $C(x, w) = 0$, output $\perp$. – If σ does not open dig to x at index i, output $\perp$. – If $i \neq 1$ and $\mathsf{Verify}(j, \mathsf{dig}, i - 1, z_{i-1}) = 0$, output $\perp$. – If $j = \mathsf{F}(k_{\mathsf{sel}}, (\mathsf{dig}, i))$, output $\perp$. – Compute $k_j \leftarrow \mathsf{F}(k, j)$. – Output $z = \mathsf{F}(k_j, (\mathsf{dig}, i))$.	– Compute $k_j \leftarrow \mathsf{F}(k, j)$. – If $\mathsf{G}(z_i) = \mathsf{G}(\mathsf{F}(k_j, (\mathsf{dig}, i)))$, output 1. – Output 0.

First, observe that $\mathsf{AggProof}$ never outputs a proof for (dig, i^*) if x_{i^*} is false (and dig is statistically binding on location i^*). Thus, in the adaptive soundness argument, the challenger can switch from checking that x_{i^*} is false to checking that $j = \mathsf{F}(k_{\mathsf{sel}}, (\mathsf{dig}, i^*))$. The soundness proof then proceeds in a sequence of hybrids for every $t = i^*, i^* + 1, \ldots, T$. In Hyb_t, the challenger defines the programs as follows:

$\mathsf{AggProof}_{i^*,t}(j, \mathsf{dig}, i, x, w, \sigma, z_{i-1})$	$\mathsf{Verify}_{i^*,t}(j, \mathsf{dig}, i, z_i)$
– If $C(x, w) = 0$, output $\perp$. – If σ does not open dig to x at index i, output $\perp$. – If $i \neq 1$ and $\mathsf{Verify}_{i^*,t}(j, \mathsf{dig}, i - 1, z_{i-1}) = 0$, output $\perp$. – If $j = \mathsf{F}(k_{\mathsf{sel}}, (\mathsf{dig}, i))$, output $\perp$. – Compute $k_j \leftarrow \mathsf{F}(k, j)$. – Output $z = \mathsf{F}(k_j, (\mathsf{dig}, i))$.	– If $i^* \leq i \leq t$ and $j = \mathsf{F}(k_{\mathsf{sel}}, (\mathsf{dig}, i^*))$: output $\perp$. – Compute $k_j \leftarrow \mathsf{F}(k, j)$. – If $\mathsf{G}(z_i) = \mathsf{G}(\mathsf{F}(k_j, (\mathsf{dig}, i)))$, output 1. – Output 0.

In the final step, to move from Hyb_{t-1} to Hyb_t, we appeal to pseudorandomness of the PRG to switch *all* of the re-randomized instances which are on-path challenges for $(\cdot, t)$ into random strings. At this point, since the PRG is expanding, there no longer exist on-path proofs for *any* statement at index t. This statistical guarantee in turn allows us to argue that there no longer exist proofs for any subsequent index. We provide the full details in Sect. 6.

While this construction does not achieve better properties than our above approach relying on homomorphism, we believe that it provides an alternative view for constructing adaptively-sound SNARGs for batch NP. Moreover, we believe the techniques are of independent interest, and may be more amenable to generalizing beyond batch NP (e.g., to monotone-policy batch NP [10,35]).

The homomorphic aggregation approach critically assumes that the proofs themselves are algebraic objects and satisfy some homomorphism. While initial constructions such as [40,42] have this property, it is not true of all adaptively-sound SNARGs (e.g., the very recent work [41]). The chaining approach does not rely on any assumption about the structure of the proofs themselves, and thus, could plausibly be based on unstructured assumptions (similar to how [41] constructs the SNARG).

2 Preliminaries

Throughout this work, we write λ to denote the security parameter. We write $\mathsf{poly}(\lambda)$ to denote a *fixed* polynomial in the security parameter λ. We say a function $f(\lambda)$ is negligible in λ if $f(\lambda) = o(\lambda^{-c})$ for all constants $c \in \mathbb{N}$ and denote this by writing $f(\lambda) = \mathsf{negl}(\lambda)$. When $x, y \in \{0,1\}^n$, we will view x and y as both bit strings of length n as well as the binary representation of an integer between 0 and $2^n - 1$. We write "$x \leq y$" to refer to the comparison

of the integer representations of x and y. We say an algorithm is efficient if it runs in probabilistic polynomial time in the length of its input. For a function $f\colon \mathcal{X} \to \mathcal{Y}$, we write $\mathsf{Im}(f)$ to denote the image of f.

Our construction will rely on sub-exponential hardness assumptions, so we will formulate some of our security definitions using (t, ε)-notation. Generally, we say that a primitive is (t, ε)-secure if, for all adversaries $\mathcal{A}$ running in time at most $t(\lambda) \cdot \mathsf{poly}(\lambda)$, there exists $\lambda_{\mathcal{A}} \in \mathbb{N}$ such that for all $\lambda \geq \lambda_{\mathcal{A}}$, the adversary's advantage is bounded by $\varepsilon(\lambda)$. We say a primitive is polynomially secure if it is $(1, \mathsf{negl}(\lambda))$-secure for some negligible function $\mathsf{negl}(\cdot)$. We now recall the main cryptographic primitives we use in this work.

Definition 1 (Indistinguishability Obfuscation [2]). *An indistinguishability obfuscator for Boolean circuits is an efficient algorithm $i\mathcal{O}(\cdot,\cdot,\cdot)$ with the following properties:*

- ***Correctness.*** *For any security parameter $\lambda \in \mathbb{N}$, circuit size parameter $s \in \mathbb{N}$, Boolean circuit C of size at most s, and input x,*

$$\Pr[\hat{C}(x) = C(x) : \hat{C} \leftarrow i\mathcal{O}(1^\lambda, 1^s, C)] = 1.$$

- ***Security.*** *For a security parameter λ and a bit $b \in \{0,1\}$, we define the program indistinguishability game between an adversary $\mathcal{A}$ and a challenger as follows:*
  - *On input security parameter 1^λ, $\mathcal{A}$ outputs a size parameter 1^s and two Boolean circuits C_0, C_1 of size at most s.*
  - *If there exists an input x such that $C_0(x) \neq C_1(x)$, then the challenger halts with output $\bot$. Otherwise, the challenger replies with $i\mathcal{O}(1^\lambda, 1^s, C_b)$.*
  - *$\mathcal{A}$ outputs a bit $b' \in \{0,1\}$, which is the output of the experiment.*

  We say that $i\mathcal{O}$ is (t, ε)-secure if for all adversaries $\mathcal{A}$ running in time at most $t(\lambda) \cdot \mathsf{poly}(\lambda)$, there exists $\lambda_{\mathcal{A}} \in \mathbb{N}$ such that for all $\lambda \geq \lambda_{\mathcal{A}}$, we have that

$$\mathsf{iOAdv}_{\mathcal{A}}(\lambda) \coloneqq |\Pr[b' = 1 : b = 1] - \Pr[b' = 1 : b = 0]| \leq \varepsilon(\lambda).$$

Definition 2 (Puncturable PRF [8,9,30]). *A puncturable pseudorandom function consists of a tuple of efficient algorithms $\Pi_{\mathsf{PPRF}} = (\mathsf{Setup}, \mathsf{Eval}, \mathsf{Puncture})$ with the following syntax:*

- $\mathsf{Setup}(1^\lambda, 1^{\ell_{\mathsf{in}}}, 1^{\ell_{\mathsf{out}}}) \to k$*: On input security parameter 1^λ, input length $1^{\ell_{\mathsf{in}}}$, and output length $1^{\ell_{\mathsf{out}}}$, the randomized setup algorithm outputs a key k. We assume that the key k contains an implicit description of ℓ_{in} and ℓ_{out}.*
- $\mathsf{Eval}(k, x) \to y$*: On input key k and point $x \in \{0,1\}^\ell_{\mathsf{in}}$, the deterministic evaluation algorithm outputs a value $y \in \{0,1\}^\ell_{\mathsf{out}}$.*
- $\mathsf{Puncture}(k, x^*) \to k^{(x^*)}$*: On input key k and point $x^* \in \{0,1\}^\ell_{\mathsf{in}}$, the puncturing algorithm outputs a punctured key $k^{(x^*)}$. We assume that the punctured key $k^{(x^*)}$ also contains an implicit description of ℓ_{in} and ℓ_{out}.*

We require that Π_{PPRF} satisfy the following properties:

– ***Punctured correctness.*** *For all* $\lambda, \ell_{\mathsf{in}}, \ell_{\mathsf{out}} \in \mathbb{N}$*, and all distinct points* $x \neq x^* \in \{0,1\}^{\ell}_{\mathsf{in}}$*,*

$$\Pr\left[\mathsf{Eval}(k,x) = \mathsf{Eval}(k^{(x^*)},x) : \begin{array}{l} k \leftarrow \mathsf{Setup}(1^\lambda, 1^{\ell_{\mathsf{in}}}, 1^{\ell_{\mathsf{out}}}) \\ k^{(x^*)} \leftarrow \mathsf{Puncture}(k, x^*) \end{array}\right] = 1.$$

– ***Puncturing Security.*** *For a security parameter* λ *and a bit* $b \in \{0,1\}$*, we define the (selective) puncturing security game between an adversary* $\mathcal{A}$ *and a challenger as follows:*

- *On input security parameter* 1^λ*,* $\mathcal{A}$ *outputs the input length* $1^{\ell_{\mathsf{in}}}$*, the output length* $1^{\ell_{\mathsf{out}}}$*, and commits to a point* $x^* \in \{0,1\}^{\ell}_{\mathsf{in}}$*.*
- *The challenger samples* $k \leftarrow \mathsf{Setup}(1^\lambda, 1^{\ell_{\mathsf{in}}}, 1^{\ell_{\mathsf{out}}})$ *and gives the punctured key* $k^{(x^*)} \leftarrow \mathsf{Puncture}(k, x^*)$ *to* $\mathcal{A}$*.*
- *If* $b = 0$*, the challenger sends* $y^* \leftarrow \mathsf{Eval}(k, x^*)$ *to* $\mathcal{A}$*. If* $b = 1$*, then it sends* $y^* \xleftarrow{R} \{0,1\}^{\ell}_{\mathsf{out}}$ *to* $\mathcal{A}$*.*
- $\mathcal{A}$ *outputs a bit* $b' \in \{0,1\}$*, which is the output of the experiment.*

We say that Π_{PPRF} *satisfes* (t,ε)*-puncturing security if for all adversaries* $\mathcal{A}$ *running in time at most* $t(\lambda) \cdot \mathsf{poly}(\lambda)$*, there exists* $\lambda_{\mathcal{A}} \in \mathbb{N}$ *such that for all* $\lambda \geq \lambda_{\mathcal{A}}$*, it holds that*

$$\mathsf{PPRFAdv}_{\mathcal{A}}(\lambda) \coloneqq |\Pr[b' = 1 : b = 1] - \Pr[b' = 1 : b = 0]| \leq \varepsilon(\lambda).$$

Definition 3 (Somewhere Extractable Hash Family [14,22]). *A somewhere extractable hash family consists of a tuple of efficient algorithms* $\Pi_{\mathsf{SEH}} = (\mathsf{Setup}, \mathsf{SetupTD}, \mathsf{Hash}, \mathsf{Open}, \mathsf{Verify}, \mathsf{Extract})$ *with the following syntax:*

– $\mathsf{Setup}(1^\lambda, 1^\ell) \to \mathsf{hk}$*: On input security parameter* 1^λ *and block size* 1^ℓ*, the setup algorithm outputs hash key* hk*.*

– $\mathsf{SetupTD}(1^\lambda, 1^\ell, i) \to (\mathsf{hk}, \mathsf{td})$*: On input security parameter* 1^λ*, block size* 1^ℓ*, and index* $i \in [2^\lambda]$*, the setup algorithm outputs hash key* hk *and trapdoor* td*.*

– $\mathsf{Hash}(\mathsf{hk}, (x_1, \ldots, x_t)) \to \mathsf{dig}$*: On input hash key* hk *and ordered list of inputs* $x_1, \ldots, x_t \in \{0,1\}^\ell$*, the hash algorithm outputs a hash value* dig*.*

– $\mathsf{Open}(\mathsf{hk}, (x_1, \ldots, x_t), i) \to \sigma$*: On input hash key* hk*, ordered list of inputs* $x_1, \ldots, x_t \in \{0,1\}^\ell$*, and index* $i \in [t]$*, the opening algorithm outputs an opening* σ*.*

– $\mathsf{Verify}(\mathsf{hk}, \mathsf{dig}, i, x, \sigma) \to 0/1$*: On input hash key* hk*, hash value* dig*, index* i*, string* $x \in \{0,1\}^\ell$*, and opening* σ*, the verification algorithm outputs a bit.*

We require that Π_{SEH} *satisfy the following properties:*

– ***Opening Completeness.*** *For any* $\lambda, \ell, t \in \mathbb{N}$ *with* $t \leq 2^\lambda$*, any* $i \in [t]$*, and any* $x_1, \ldots, x_t \in \{0,1\}^\ell$*,*

$$\Pr\left[\mathsf{Verify}(\mathsf{hk}, \mathsf{dig}, i, x_i, \sigma) = 1 : \begin{array}{l} \mathsf{hk} \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell) \\ \mathsf{dig} = \mathsf{Hash}(\mathsf{hk}, (x_1, \ldots, x_t)) \\ \sigma = \mathsf{Open}(\mathsf{hk}, (x_1, \ldots, x_t), i) \end{array}\right] = 1.$$

– ***Succinctness.*** *There exists a fixed polynomial* p *such that the lengths of the hash values* dig *output by* Hash *and the lengths of the openings* σ *output by* Open *in the completeness experiment satisfy* $|\mathsf{dig}|, |\sigma| = p(\lambda, \ell)$*.*

– ***Index hiding.*** *For a security parameter λ and a bit $b \in \{0,1\}$, we define the index-hiding security game between an adversary $\mathcal{A}$ and a challenger as follows:*
  - *On input security parameter 1^λ, algorithm $\mathcal{A}$ outputs the block length 1^ℓ and an index $i \in [2^\lambda]$.*
  - *If $b = 0$, the challenger samples $\mathsf{hk} \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell)$. If $b = 1$, it samples $(\mathsf{hk}, \mathsf{td}) \leftarrow \mathsf{SetupTD}(1^\lambda, 1^\ell, i)$ The challenger sends hk to $\mathcal{A}$.*
  - *$\mathcal{A}$ outputs a bit $b' \in \{0,1\}$, which is the output of the experiment.*

  We say that Π_{SEH} satisfies (t, ε)-index-hiding security if for all adversaries $\mathcal{A}$ running in time $t(\lambda) \cdot \mathsf{poly}(\lambda)$, there exists $\lambda_{\mathcal{A}} \in \mathbb{N}$ such that for all $\lambda \geq \lambda_{\mathcal{A}}$,

$$\mathsf{SEHAdv}_{\mathcal{A}}(\lambda) \coloneqq |\Pr[b' = 1 : b = 1] - \Pr[b' = 1 : b = 0]| \leq \varepsilon(\lambda).$$

– ***Extraction Correctness.*** *For any $\lambda, \ell, t \in \mathbb{N}$ with $t \leq 2^\lambda$, any $i \in [t]$, any $x_1, \ldots, x_t \in \{0,1\}^\ell$,*

$$\Pr\left[x_i \neq \mathsf{Extract}(\mathsf{td}, \mathsf{dig}, i) : \begin{matrix}(\mathsf{hk}, \mathsf{td}) \leftarrow \mathsf{SetupTD}(1^\lambda, 1^\ell, i) \\ \mathsf{dig} = \mathsf{Hash}(\mathsf{hk}, (x_1, \ldots, x_t))\end{matrix}\right] = 0.$$

– ***Statistically Binding.*** *For any $\lambda, \ell, t \in \mathbb{N}$ with $t \leq 2^\lambda$, any $i \in [t]$,*

$$\Pr\left[\begin{matrix}\exists \mathsf{dig}, x, \sigma : x \neq \mathsf{Extract}(\mathsf{td}, \mathsf{dig}, i) \\ \wedge\ \mathsf{Verify}(\mathsf{hk}, \mathsf{dig}, i, x, \sigma) = 1\end{matrix} : (\mathsf{hk}, \mathsf{td}) \leftarrow \mathsf{SetupTD}(1^\lambda, 1^\ell, i)\right] = 0.$$

2.1 Batch Arguments for NP

Definition 4 (Circuit Satisfiability). *We define the Boolean circuit satisfiability language $\mathcal{L}_{\mathsf{SAT}}$ as follows:*

$$\mathcal{L}_{\mathsf{SAT}} = \{(C, x) \mid \exists w \in \{0,1\}^v \text{ s.t. } C(x, w) = 1\}$$

where C is a Boolean circuit $C\colon \{0,1\}^n \times \{0,1\}^v \to \{0,1\}$ and $x \in \{0,1\}^n$ is a statement.

Definition 5 (Non-interactive Batch Argument for NP). *A non-interactive batch argument (*BARG*) for the Boolean circuit satisfiability language $\mathcal{L}_{\mathsf{SAT}}$ is a tuple of efficient algorithms $\Pi_{\mathsf{BARG}} = (\mathsf{Setup}, \mathsf{P}, \mathsf{V})$ with the following syntax:*

– $\mathsf{Setup}(1^\lambda, T, C) \to \mathsf{crs}$*: On input security parameter 1^λ, batch size T, and Boolean circuit C, the setup algorithm outputs a common reference string* crs*.*
– $\mathsf{P}(\mathsf{crs}, (x_1, \ldots, x_T), (w_1, \ldots, w_T)) \to \pi$*: On input common reference string* crs*, statements $x_1, \ldots, x_T$, and witnesses $w_1, \ldots, w_T$, the prover algorithm outputs a proof π.*
– $\mathsf{V}(\mathsf{crs}, (x_1, \ldots, x_T), \pi) \to b$*: On input common reference string* crs*, statements $x_1, \ldots, x_T$, and proof π, the verifier algorithm outputs a bit $b \in \{0,1\}$.*

We require that Π_{BARG} *satisfy the following properties:*

- ***Completeness.*** *For any security parameter* $\lambda \in \mathbb{N}$*, polynomials* $n = n(\lambda), v = v(\lambda), T = T(\lambda)$*, Boolean circuit* $C\colon \{0,1\}^n \times \{0,1\}^v \to \{0,1\}$ *of* $\mathsf{poly}(\lambda)$ *size, and statements* $x_1, \ldots, x_T \in \{0,1\}^n$ *and witnesses* $w_1, \ldots, w_T \in \{0,1\}^v$ *such that* $C(x_i, w_i) = 1$ *for all* $i \in [T]$*, it holds that*

$$\Pr\left[\mathsf{V}(\mathsf{crs}, (x_1, \ldots, x_T), \pi) = 1 : \begin{array}{l} \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, T, C) \\ \pi \leftarrow \mathsf{P}(\mathsf{crs}, (x_1, \ldots, x_T), (w_1, \ldots, w_T)) \end{array}\right] = 1.$$

- ***Succinctness.*** *There exists a universal polynomial* p *such that in the completeness experiment above, we have that* $|\pi| = p(\lambda, \log T, |C|)$*. We say the proof is* fully succinct *if we have that* $|\pi| = p(\lambda, \log T, \log |C|)$*.*[4]
- ***Adaptive Soundness.*** *For a security parameter* λ*, we define the adaptive soundness game between an adversary* $\mathcal{A}$ *and a challenger as follows:*
  - *On input security parameter* 1^λ*,* $\mathcal{A}$ *starts by outputting a Boolean circuit* $C\colon \{0,1\}^n \times \{0,1\}^v \to \{0,1\}$ *and a number of instances* T*.*
  - *The challenger replies with* $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, T, C)$*.*
  - $\mathcal{A}$ *outputs statements* $x_1, \ldots, x_T \in \{0,1\}^n$ *and a proof* π*.*
  - *The output of the experiment is* $b = 1$ *if there exists some* $i \in [T]$ *such that* $(C, x_i) \notin \mathcal{L}_{\mathsf{SAT}}$ *and* $\mathsf{Verify}(\mathsf{crs}, (x_1, \ldots, x_T), \pi) = 1$ *and* $b = 0$ *otherwise.*

  We say that Π_{BARG} *is adaptively sound if for all efficient adversaries* $\mathcal{A}$*, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for all* $\lambda \in \mathbb{N}$*,* $\Pr[b = 1] \leq \mathsf{negl}(\lambda)$ *in the adaptive soundness game.*

Remark 1 (Supporting Arbitrary Batch Size). In our definition, the Setup algorithm needs to take the batch size T as input (in binary). Note that this restriction can be generically removed using a standard "powers-of-two" construction, where we generate a CRS for every value of $T = 2^i$ for $i \in [\lambda]$. This is still efficient as the size of each CRS depends only polylogarithmically on the batch size, and padding to the next power of two only incurs constant overhead.

Definition 6 (Perfect Zero-knowledge). *A* BARG $\Pi_{\mathsf{BARG}} = (\mathsf{Setup}, \mathsf{P}, \mathsf{V})$ *for the Boolean circuit satisfiability language* $\mathcal{L}_{\mathsf{SAT}}$ *satisfies perfect zero-knowledge if there exists an efficient simulator* $\mathcal{S} = (\mathcal{S}_0, \mathcal{S}_1)$ *such that for any Boolean circuit* $C\colon \{0,1\}^n \times \{0,1\}^v \to \{0,1\}$ *and any statements* $X = (x_1, \ldots, x_T)$ *and witnesses* $W = (w_1, \ldots, w_T)$ *such that* $C(x_i, w_i) = 1$ *for all* $i \in [T]$*, the following distributions are identical:*

$$\left\{(\mathsf{crs}, X, \pi) : \begin{array}{l} \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, T, C) \\ \pi \leftarrow \mathsf{P}(\mathsf{crs}, X, W) \end{array}\right\} \equiv \left\{(\mathsf{crs}, X, \pi) : \begin{array}{l} (\mathsf{crs}, \mathsf{st}) \leftarrow \mathcal{S}_0(1^\lambda, T, C) \\ \pi \leftarrow \mathcal{S}_1(\mathsf{st}, X) \end{array}\right\}.$$

2.2 Discrete Log Assumptions

Notation. For a positive integer $p > 1$, we write $\mathbb{Z}_p$ to denote the set of integers $\{0, \ldots, p-1\}$. We write $\mathbb{Z}_p^*$ to denote the multiplicative group of integers modulo p.

[4] This is the notion of succinctness that our constructions achieve.

Definition 7 (Prime-Order Group Generator). *Let λ be a security parameter. A prime-order group generator is an efficient algorithm* $\mathsf{GroupGen}$ *that takes as input security parameter 1^λ and outputs the description $\mathcal{G} = (\mathbb{G}, p, g)$ of a group $\mathbb{G}$ of prime order $p = 2^{\Theta(\lambda)}$ and generated by $g \in \mathbb{G}$. Moreover, we require that the group operation in $\mathbb{G}$ be efficiently computable.*

Definition 8 (Discrete Log Assumption). *Let* $\mathsf{GroupGen}$ *be a prime-order group generator. We say that the discrete log assumption holds with respect to* $\mathsf{GroupGen}$ *if for all efficient adversaries $\mathcal{A}$, there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that for all $\lambda \in \mathbb{N}$,*

$$\Pr[\mathcal{A}(1^\lambda, \mathcal{G}, g^x) = x : \mathcal{G} = (\mathbb{G}, p, g) \leftarrow \mathsf{GroupGen}(1^\lambda), x \xleftarrow{R} \mathbb{Z}_p] \leq \mathsf{negl}(\lambda).$$

Definition 9 (Decisional Diffie-Hellman Assumption). *For a security parameter λ, a bit $b \in \{0,1\}$, and a prime-order group generator* $\mathsf{GroupGen}$*, we define the decisional Diffie-Hellman (*DDH*) security game between an adversary $\mathcal{A}$ and a challenger as follows:*

- *The challenger starts by sampling $\mathcal{G} = (\mathbb{G}, g, p) \leftarrow \mathsf{GroupGen}(1^\lambda)$ and $x, y \xleftarrow{R} \mathbb{Z}_p^*$.*
- *If $b = 0$, the challenger computes $z = xy \in \mathbb{Z}_p^*$. If $b = 1$, the challenger samples $z \xleftarrow{R} \mathbb{Z}_p^*$.*
- *The challenger then sends $(\mathcal{G}, g^x, g^y, g^z)$ to $\mathcal{A}$.*
- *$\mathcal{A}$ outputs a bit b', which is the output of the experiment.*

We say that the DDH *assumption holds with respect to* $\mathsf{GroupGen}$ *if for all efficient adversaries $\mathcal{A}$, there exists $\lambda_\mathcal{A} \in \mathbb{N}$ such that for all security parameters $\lambda \geq \lambda_\mathcal{A}$, it holds that*

$$\mathsf{DDHAdv}_\mathcal{A}(\lambda) \coloneqq |\Pr[b' = 1 : b = 0] - \Pr[b' = 1 : b = 1]| \leq \varepsilon(\lambda)$$

in the DDH *security game.*

3 Homomorphic Re-randomizable OWFs

In this section, we introduce the notion of a homomorphic re-randomizable OWF, which is one of the main building blocks we use in our construction of an adaptive fully succinct BARG in Sect. 4. Then, in Sect. 3.1, we show how to construct a homomorphic re-randomizable OWF from discrete log.

Definition 10 (Homomorphic Re-randomizable OWF). *A homomorphic re-randomizable OWF is a tuple of efficient algorithms $\Pi_{\mathsf{OWF}} =$* $(\mathsf{Setup}, \mathsf{Eval}, \mathsf{Rerandomize}, \mathsf{InHom}, \mathsf{OutHom}, \mathsf{RecoverSolution})$ *with the following syntax:*

- $\mathsf{Setup}(1^\lambda, 1^m) \to \mathsf{crs}$: *On input security parameter 1^λ and rerandomization parameter 1^m, the setup algorithm outputs a common reference string* crs*. We assume that the* crs *contains an implicit description of the input space $\mathcal{Z}$ and the output space $\mathcal{Y}$ of* $\mathsf{Eval}(\mathsf{crs}, \cdot)$*.*

- $\mathsf{Eval}(\mathsf{crs}, z) \to y$: *On input the common reference string* crs *and a preimage* $z \in \mathcal{Z}$*, the deterministic evaluation algorithm outputs challenge* $y \in \mathcal{Y}$*.*
- $\mathsf{Rerandomize}(\mathsf{crs}, y) \to y'$: *On input the common reference string* crs *and a challenge* $y \in \mathcal{Y}$*, the randomization algorithm outputs a new challenge* $y' \in \mathcal{Y}$*.*
- $\mathsf{RecoverSolution}(\mathsf{crs}, z', r) \to z$: *On input the common reference string* crs*, a preimage* $z' \in \mathcal{Z}$*, and randomness* r*, the solution recovery algorithm outputs a new preimage* $z \in \mathcal{Z}$*.*
- $\mathsf{InHom}(\mathsf{crs}, (z_1, \ldots, z_\ell)) \to z$*: On input the common reference string* crs *and preimages* $z_1, \ldots, z_\ell \in \mathcal{Z}$*, the input homomorphism algorithm outputs a new preimage* $z \in \mathcal{Z}$*.*
- $\mathsf{OutHom}(\mathsf{crs}, (y_1, \ldots, y_\ell)) \to y$*: On input the common reference string* crs *and challenges* $y_1, \ldots, y_\ell \in \mathcal{Y}$*, the output homomorphism algorithm outputs a new challenge* $y \in \mathcal{Y}$*.*

We require that Π_{OWF} *satisfy the following properties:*

- ***Homomorphism.*** *For all* $\lambda, m \in \mathbb{N}$*, all* crs *in the support of* $\mathsf{Setup}(1^\lambda, 1^m)$*, all* $z_1, \ldots, z_\ell \in \mathcal{Z}$*, we have that*

$$\Pr[\mathsf{Eval}(\mathsf{crs}, \mathsf{InHom}(\mathsf{crs}, (z_1, \ldots, z_\ell))) = \mathsf{OutHom}(\mathsf{crs}, (y_1, \ldots, y_\ell))] = 1,$$

  where $y_i = \mathsf{Eval}(\mathsf{crs}, z_i)$ *for all* $i \in [\ell]$*. Further,* InHom *has a corresponding inversion algorithm* InHom^{-1} *such that for all* crs *in the support of* $\mathsf{Setup}(1^\lambda, 1^m)$*, for all preimages* $z, z' \in \mathcal{Z}$ *and challenges* $y, y' \in \mathsf{Im}(\mathsf{Eval}(\mathsf{crs}, \cdot))$ *such that*

$$\mathsf{Eval}(\mathsf{crs}, z) = \mathsf{OutHom}(\mathsf{crs}, (y, y')) \text{ and } \mathsf{Eval}(\mathsf{crs}, z') = y',$$

  we have that

$$\mathsf{Eval}(\mathsf{crs}, \mathsf{InHom}^{-1}(\mathsf{crs}, (z, z'))) = y.$$

- ***One-wayness.*** *For a security parameter* λ*, a rerandomization parameter* m*, and a bit* $b \in \{0, 1\}$*, we define the one-wayness security game between an adversary* $\mathcal{A}$ *and a challenger as follows:*
  - *The challenger samples* $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^m)$*. The challenger samples* $z \xleftarrow{R} \mathcal{Z}$ *and computes* $y \leftarrow \mathsf{Eval}(\mathsf{crs}, z)$*.*
  - *The challenger then sends* (crs, y) *to* $\mathcal{A}$*.*
  - *Algorithm* $\mathcal{A}$ *sends a preimage* z' *to the challenger.*
  - *The challenger outputs a bit* $b' \in \{0, 1\}$ *where* $b' = 1$ *if and only if* $\mathsf{Eval}(\mathsf{crs}, z') = y$*.*

  We say that Π_{OWF} *is* (t, ε)*-one-way if for all polynomials* $m = m(\lambda)$ *and all adversaries* $\mathcal{A}$ *running in time at most* $t(\lambda) \cdot \mathsf{poly}(\lambda)$*, there exists* $\lambda_{\mathcal{A}} \in \mathbb{N}$ *such that for all security parameters* $\lambda \geq \lambda_{\mathcal{A}}$*, it holds that*

$$\mathsf{PRGAdv}_{\mathcal{A}}(\lambda) := \Pr[b' = 1] \leq \varepsilon(\lambda)$$

  in the one-wayness security game.

– ***Rerandomization Correctness.*** *For all* $\lambda \in \mathbb{N}$*, all polynomials* $m = m(\lambda)$*, all* crs *in the support of* $\mathsf{Setup}(1^\lambda, 1^m)$*, all preimages* $z' \in \mathcal{Z}$*, all* $y \in \mathsf{Im}(\mathsf{Eval}(\mathsf{crs}, \cdot))$*, and all randomness* r *such that*

$$\mathsf{Eval}(\mathsf{crs}, z') = \mathsf{Rerandomize}(\mathsf{crs}, y; r)$$

we have that

$$\mathsf{Eval}(\mathsf{crs}, \mathsf{RecoverSolution}(\mathsf{crs}, z', r)) = y.$$

– ***Rerandomization Security.*** *For a security parameter* λ*, a rerandomization parameter* m*, and a bit* $b \in \{0,1\}$*, we define the rerandomization security game between an adversary* $\mathcal{A}$ *and a challenger as follows:*

- *The challenger samples* $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^m)$*,* $z_{\mathsf{base}} \xleftarrow{R} \mathcal{Z}$*, and* $y_{\mathsf{base}} \leftarrow \mathsf{Eval}(\mathsf{crs}, z_{\mathsf{base}})$
- *If* $b = 0$*, the challenger samples* $z^* \xleftarrow{R} \mathcal{Z}$*,* $y^* \leftarrow \mathsf{Eval}(\mathsf{crs}, z^*)$*. If* $b = 1$*, the challenger samples* $y^* \leftarrow \mathsf{Rerandomize}(\mathsf{crs}, y_{\mathsf{base}})$*.*
- *The challenger then sends* $(\mathsf{crs}, y_{\mathsf{base}}, y^*)$ *to* $\mathcal{A}$*.*
- $\mathcal{A}$ *outputs a bit* b'*, which is the output of the experiment.*

We say that Π_{OWF} *satisfies* (t, ε)*-rerandomization security if for all polynomials* $m = m(\lambda)$ *and all adversaries* $\mathcal{A}$ *running in time at most* $t(\lambda) \cdot \mathsf{poly}(\lambda)$*, there exists* $\lambda_{\mathcal{A}} \in \mathbb{N}$ *such that for all security parameters* $\lambda \geq \lambda_{\mathcal{A}}$*, it holds that*

$$\mathsf{RerandAdv}_{\mathcal{A}}(\lambda) := |\Pr[b' = 1 : b = 0] - \Pr[b' = 1 : b = 1]| \leq \varepsilon(\lambda)$$

in the rerandomization security game.

– ***Succinctness.*** *There exists a polynomial* p *such that for all* $\lambda, m \in \mathbb{N}$ *and all* crs *in the support of* $\mathsf{Setup}(1^\lambda, 1^m)$*, it holds that the input length of* $\mathsf{Eval}(\mathsf{crs}, \cdot)$ *is* $|z| \leq p(\lambda + \log m)$*.*

3.1 Constructing Homomorphic Re-randomizable OWFs

In this section, we show that the construction of a re-randomizable OWF from discrete log [40, Construction 5.3] is a homomorphic re-randomizable OWF. Though we do not formalize it in this work, the second construction of a re-randomizable OWF in [40] based on computing modular square roots (i.e., the function $f(z) = z^2 \mod N$ where $N = pq$ is an RSA modulus) is also homomorphic, as $f(z_0 z_1) = (z_0 z_1)^2 = z_0^2 z_1^2 = f(z_0) f(z_1) \mod N$. We start by recalling the discrete-log-based construction from [40], with the addition of the InHom and OutHom algorithms for the homomorphism property. For simplicity, we describe the scheme with *additive* blinding rather than multiplicative blinding:

Construction 1 (Homomorphic Re-randomizable OWF). *Let* $\mathsf{GroupGen}$ *be a prime-order group generator. We construct a homomorphic re-randomizable OWF* $\Pi_{\mathsf{OWF}} = (\mathsf{Setup}, \mathsf{Eval}, \mathsf{Rerandomize}, \mathsf{InHom}, \mathsf{OutHom}, \mathsf{RecoverSolution})$ *as follows:*

– $\mathsf{Setup}(1^\lambda, 1^m)$: *On input the security parameter* 1^λ *and rerandomization parameter* 1^m*, the setup algorithm samples* $\mathcal{G} = (\mathbb{G}, p, g) \leftarrow \mathsf{GroupGen}(1^\lambda)$ *and outputs* $\mathsf{crs} = \mathcal{G}$*. The domain of the OWF is* $\mathcal{Z} = \mathbb{Z}_p$ *and the range is* $\mathcal{Y} = \mathbb{G}$*.*

- $\mathsf{Eval}(\mathsf{crs}, z) \to y$: *On input the common reference string* crs *and a preimage* $z \in \mathbb{Z}_p$*, the evaluation algorithm outputs* $g^z \in \mathbb{G}$*.*
- $\mathsf{Rerandomize}(\mathsf{crs}, y) \to y'$: *On input the common reference string* crs *and a challenge* $y \in \mathbb{G}$*, the randomization algorithm samples* $r \xleftarrow{R} \mathbb{Z}_p$ *and outputs* $y \cdot g^r \in \mathbb{G}$*.*
- $\mathsf{RecoverSolution}(\mathsf{crs}, z', r) \to z$: *On input the common reference string* crs*, a preimage* $z' \in \mathbb{Z}_p$*, and randomness* $r \in \mathbb{Z}_p$*, the solution recovery algorithm outputs* $z' - r \in \mathbb{Z}_p$*.*
- $\mathsf{InHom}(\mathsf{crs}, (z_1, \ldots, z_\ell)) \to z$*: On input common reference string* crs *and OWF preimages* $z_1, \ldots, z_\ell \in \mathbb{Z}_p$*, the input homomorphism outputs* $\sum_{i\in[\ell]} z_i \in \mathbb{Z}_p$*.*
- $\mathsf{OutHom}(\mathsf{crs}, (y_1, \ldots, y_\ell)) \to y$*: On input common reference string* crs *and challenges* $y_1, \ldots, y_\ell \in \mathbb{G}$*, the output homomorphism outputs* $\prod_{i\in[\ell]} y_i \in \mathbb{G}$*.*

We refer to [40, §5.1] for the proofs of the one-wayness, re-randomization correctness, re-randomization security, and succinctness properties. Here, we show the homomorphism property.

Theorem 2 (Homomorphism). *Construction 1 satisfies homomorphism.*

Proof. For any $\mathsf{crs} = (\mathbb{G}, p, g)$ in the support of $\mathsf{GroupGen}(1^\lambda)$, and any $z_1, \ldots, z_\ell \in \mathbb{Z}_p$, we have that

$$\mathsf{Eval}(\mathsf{crs}, \mathsf{InHom}(\mathsf{crs}, \{z_i\}_{i\in[\ell]}) = g^{\sum_{i\in[\ell]} z_i} = \prod_{i\in[\ell]} g^{z_i} = \mathsf{OutHom}(\mathsf{crs}, \{\mathsf{Eval}(\mathsf{crs}, z_i)\}_{i\in[\ell]}).$$

Define $\mathsf{InHom}^{-1}(\mathsf{crs}, z, z') = z - z'$. Then, for any $z, z' \in \mathbb{Z}_p^*$ and $y, y' \in \mathbb{G}$, if

$$g^z = \mathsf{Eval}(\mathsf{crs}, z) = \mathsf{OutHom}(\mathsf{crs}, y, y') = yy' \text{ and } g^{z'} = \mathsf{Eval}(\mathsf{crs}, z') = y',$$

then

$$\mathsf{Eval}(\mathsf{crs}, \mathsf{InHom}^{-1}(\mathsf{crs}, z, z')) = g^{z-z'} = \frac{yy'}{y'} = y. \square$$

4 SNARG for Batch NP from Homomorphic Re-Randomizable OWFs

In this section, we show how to construct a fully succinct SNARG for batch NP using $i\mathcal{O}$ together with a homomorphic re-randomizable OWF. Our construction follows a similar template as the construction from [40].

Construction 2 (Adaptive Batch Argument for NP). *Let* λ *be a security parameter. We construct a BARG scheme that supports NP languages with an arbitrary polynomial number* $T = T(\lambda) < 2^\lambda$ *of instances of length* $n = n(\lambda)$*. Our construction will leverage sub-exponential hardness of the below primitives (except for one-wayness of* Π_{OWF}*). Our construction relies on the following primitives:*

- *Let $i\mathcal{O}$ be an indistinguishability obfuscator for Boolean circuits.*
- *Let $\Pi_{\mathsf{PPRF}} = (\mathsf{F.Setup}, \mathsf{F.Eval}, \mathsf{F.Puncture})$ be a puncturable PRF. For a key k and an input x, we will write $\mathsf{F}(k, x)$ to denote $\mathsf{F.Eval}(k, x)$.*
- *Let $\Pi_{\mathsf{OWF}} = (\mathsf{OWF.Setup}, \mathsf{OWF.Eval}, \mathsf{OWF.Rerandomize}, \mathsf{OWF.RecoverSolution}, \mathsf{OWF.InHom}, \mathsf{OWF.OutHom})$ be a homomorphic re-randomizable one-way function. For parameters crs_f and an input x, we will write $f(x)$ to denote $\mathsf{OWF.Eval}(\mathsf{crs}_f, x)$.*

We will describe how to define the polynomials $\lambda_{\mathsf{obf}}, \lambda_{\mathsf{PRF}}$, and m in the security analysis. We construct a fully succinct non-interactive batch argument $\Pi_{\mathsf{BARG}} = (\mathsf{Gen}, \mathsf{P}, \mathsf{V})$ for NP as follows:

- $\mathsf{Setup}(1^\lambda, T, C) \to \mathsf{crs}$*: On input security parameter 1^λ, batch size T, and Boolean circuit $C\colon \{0,1\}^n \times \{0,1\}^v \to \{0,1\}$, the setup algorithm does the following:*
  - *Sample OWF parameters $\mathsf{crs}_f \leftarrow \mathsf{OWF.Setup}(1^\lambda, 1^m)$.*
  - *Let $t = \log(T+1)$. Let ρ be the input length of $f(\cdot)$. Let τ be the number of bits of randomness the $\mathsf{F.Setup}(1^{\lambda_{\mathsf{PRF}}}, 1^{n+t}, 1^\rho)$ algorithm takes.*
  - *Sample a "selector" PPRF key $k_{\mathsf{sel}} \leftarrow \mathsf{F.Setup}(1^{\lambda_{\mathsf{PRF}}}, 1^{n+t}, 1^t)$.*
  - *Sample a "key generator" PPRF key $k \leftarrow \mathsf{F.Setup}(1^{\lambda_{\mathsf{PRF}}}, 1^t, 1^\tau)$.*
  - *Define the GenSol program with the circuit C and PPRF keys k, k_{sel} hard-coded:*

    $\mathsf{GenSol}[C, k, k_{\mathsf{sel}}]$

    Inputs: *index i, selection symbol j, statement x_i, witness w_i*
    1. *If $C(x_i, w_i) = 0$, output $\perp$.*
    2. *If $j = \mathsf{F}(k_{\mathsf{sel}}, (x_i, i))$, output $\perp$.*
    3. *Compute $k_j \leftarrow \mathsf{F.Setup}(1^{\lambda_{\mathsf{PRF}}}, 1^{n+t}, 1^\rho; \mathsf{F}(k, j))$.*
    4. *Output $z_i = \mathsf{F}(k_j, (x_i, i))$.*

  - *Define the $\mathsf{GenChall}$ program with the OWF parameters crs_f and PPRF key k hardcoded:*

    $\mathsf{GenChall}[\mathsf{crs}_f, k]$

    Inputs: *index i, selection symbol j, statement x_i*
    1. *Compute $k_j \leftarrow \mathsf{F.Setup}(1^{\lambda_{\mathsf{PRF}}}, 1^{n+t}, 1^\rho; \mathsf{F}(k, j))$.*
    2. *Output $f(\mathsf{F}(k_j, (x_i, i)))$.*

  - *Let $s = s(\lambda, n, |C|)$ be the maximum size of the $\mathsf{GenChall}$ and GenSol programs as well as those appearing in the security analysis.*
  - *Construct the obfuscated programs*

    $$\mathsf{ObfGenChall} \leftarrow i\mathcal{O}(1^{\lambda_{\mathsf{obf}}}, 1^s, \mathsf{GenChall}[\mathsf{crs}_f, k])$$

    and

    $$\mathsf{ObfGenSol} \leftarrow i\mathcal{O}(1^{\lambda_{\mathsf{obf}}}, 1^s, \mathsf{GenSol}[C, k, k_{\mathsf{sel}}]).$$

  - *Output* $\mathsf{crs} = (\mathsf{crs}_f, \mathsf{ObfGenChall}, \mathsf{ObfGenSol})$.
- $\mathsf{P}(\mathsf{crs}, (x_1, \ldots, x_T), (w_1, \ldots, w_T)) \to \pi = (j, z)$*:*
  - *Set* $i = 1, j = 1$.
  - *While* $i \leq T$:
    - *Compute* $z_i \leftarrow \mathsf{ObfGenSol}(i, j, x_i, w_i)$.
    - *If* $z_i = \bot$*, set* $i = 1, j = j + 1$.
    - *Else, set* $i = i + 1$.
  - *Compute* $z = \mathsf{OWF.InHom}(\mathsf{crs}_f, z_1, \ldots, z_T)$.
  - *Output* (j, z).
- $\mathsf{V}(\mathsf{crs}, (x_1, \ldots, x_T), \pi = (j, z)) \to 0/1$*:*
  - *For* $i \in [T]$*: compute* $y_i \leftarrow \mathsf{ObfGenChall}(i, j, x_i)$.
  - *Compute* $y = \mathsf{OWF.OutHom}(\mathsf{crs}_f, y_1, \ldots, y_T)$.
  - *Output 1 if and only if* $f(z) = y$.

Theorem 3 (Completeness). *Suppose* $i\mathcal{O}$ *is correct and* Π_{OWF} *satisfies homomorphism. Then Construction 2 is complete.*

Proof. Take any security parameter $\lambda \in \mathbb{N}$, any Boolean circuit $C\colon \{0,1\}^n \times \{0,1\}^v \to \{0,1\}$, any $T \leq 2^\lambda$, and any statements $(x_1, \ldots, x_T)$ and witnesses $(w_1, \ldots, w_T)$ such that $C(x_i, w_i) = 1$ for all $i \in [T]$. Let

$$\mathsf{crs} = (\mathsf{crs}_f, \mathsf{ObfGenSol}, \mathsf{ObfGenChall}) \leftarrow \mathsf{Setup}(1^\lambda, C, T)$$

and $\pi = (j, z) \leftarrow \mathsf{P}(\mathsf{crs}, (x_1, \ldots, x_T), (w_1, \ldots, w_T))$. Consider the output of $\mathsf{V}(\mathsf{crs}, (x_1, \ldots, x_T), \pi)$:

- By construction, $\mathsf{ObfGenSol}$ is an obfuscation of the program $\mathsf{GenSol}[C, k, k_{\mathsf{sel}}]$, where

  $$k_{\mathsf{sel}} \leftarrow \mathsf{F.Setup}(1^{\lambda_{\mathsf{PRF}}}, 1^{n+t}, 1^t) \quad \text{and} \quad k \leftarrow \mathsf{F.Setup}(1^{\lambda_{\mathsf{PRF}}}, 1^t, 1^\tau).$$

  Algorithm P obtains $(j, z_1), \ldots, (j, z_T)$ by evaluating $\mathsf{ObfGenSol}$ on inputs (i, j, x_i, w_i). By correctness of $i\mathcal{O}$ and the definition of GenSol, this means that $z_i = \mathsf{F}(k_j, (x_i, i))$ for all $i \in [T]$. Note that an index $j \in [T + 1]$ always exists, because for each index i, there is just a single index $j_i = \mathsf{F}(k_{\mathsf{sel}}, (x_i, i))$ where the GenSol program outputs $\bot$. Since there are at most T instances, there are at most T indices $j \in [T + 1]$ that fail, or equivalently, there must exist at least one index $j \in [T + 1]$ where $j \neq \mathsf{F}(k_{\mathsf{sel}}, (x_i, i))$ for all $i \in [T]$.
- By construction, $\mathsf{ObfGenChall}$ is an obfuscation of the program $\mathsf{GenChall}[\mathsf{crs}_f, k]$ where $\mathsf{crs}_f \leftarrow \mathsf{OWF.Setup}(1^\lambda, 1^m)$. Algorithm V computes the instance $y_i \leftarrow \mathsf{ObfGenChall}(i, j, x_i)$. By correctness of $i\mathcal{O}$ and the definition of $\mathsf{GenChall}$, this means that $y_i = f(\mathsf{F}(k_j, (x_i, i))) = f(z_i)$ for all $i \in [T]$.
- Finally, algorithm P computes $z = \mathsf{OWF.InHom}(\mathsf{crs}_f, z_1, \ldots, z_T)$ and algorithm V computes $y = \mathsf{OWF.OutHom}(\mathsf{crs}_f, y_1, \ldots, y_T)$. By homomorphism of Π_{OWF}, we have that

  $$\Pr[f(\mathsf{OWF.InHom}(\mathsf{crs}_f, z_1, \ldots, z_T)) = \mathsf{OWF.OutHom}(\mathsf{crs}_f, f(z_1), \ldots, f(z_T)] = 1 = \Pr[f(z) = y].$$

Thus V outputs 1 with probability 1.

Theorem 4 (Succinctness). *Suppose Π_{OWF} is succinct. Then Construction 2 is succinct.*

Proof. A proof (j, z) in Construction 2 consists of a selection symbol $j \in [T+1]$ and a OWF preimage z. Since Π_{OWF} is succinct, there is a fixed polynomial p such that $|z| \leq p(\lambda + \log m)$. Since $m(\lambda, n)$ in Construction 2 is a fixed polynomial in the security parameter λ and the statement length n and the statement length is always upper-bounded by the circuit size, it follows that $|\pi| \leq \mathsf{poly}(\lambda + \log |C|) + \log T$.

Theorem 5 (Adaptive Soundness). *Suppose $i\mathcal{O}$ is $(1, 2^{-\lambda_{\mathsf{obf}}^{\varepsilon_{\mathsf{obf}}}})$-secure, Π_{PPRF} satisfies punctured correctness and $(1, 2^{-\lambda_{\mathsf{PRF}}^{\varepsilon_{\mathsf{PRF}}}})$-puncturing security, and Π_{OWF} satisfies rerandomization correctness, $(1, \mathsf{negl}(\lambda))$-one-wayness, and $(1, 2^{-m^{\varepsilon_m}})$-rerandomization security for constants $\varepsilon_{\mathsf{obf}}, \varepsilon_{\mathsf{PRF}}, \varepsilon_m \in (0,1)$ and security parameters $\lambda_{\mathsf{obf}} = (\lambda+n)^{1/\varepsilon_{\mathsf{obf}}}, \lambda_{\mathsf{PRF}} = (\lambda+n)^{1/\varepsilon_{\mathsf{PRF}}}, m = (\lambda+n)^{1/\varepsilon_m}$. Then Construction 2 is adaptively sound.*

Proof. We refer to the full version for a proof of adaptive soundness.

Theorem 6 (Perfect Zero-Knowledge). *Suppose $i\mathcal{O}$ is correct. Then Construction 2 satisfies perfect zero-knowledge.*

Proof. We construct the simulator as follows:

- $\mathcal{S}_0(1^\lambda, T, C)$: On input the security parameter λ, a batch size T, and a Boolean circuit $C\colon \{0,1\}^n \times \{0,1\}^v \to \{0,1\}$, the simulator samples the common reference string $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, T, C)$ exactly as in the real scheme. Let $\mathsf{crs}_f, k_{\mathsf{sel}}, k$ be the underlying OWF parameters and PPRF keys sampled in Setup. The simulator outputs the crs along with the state $\mathsf{st} = (\mathsf{crs}_f, k_{\mathsf{sel}}, k)$.
- $\mathcal{S}_1(\mathsf{st}, (x_1, \ldots, x_T))$: On input the state $\mathsf{st} = (\mathsf{crs}_f, k_{\mathsf{sel}}, k)$ and statements $(x_1, \ldots, x_T)$, the simulator computes $j_i \leftarrow \mathsf{F}(k_{\mathsf{sel}}, (x_i, i))$ and selects the smallest $j \in [T+1]$ such that $j \neq j_i$ for all $i \in [T]$. It then computes $k_j \leftarrow \mathsf{F.Setup}(1^{\lambda_{\mathsf{PRF}}}, 1^{n+t}, 1^\rho; \mathsf{F}(k, j))$ and $z_i \leftarrow \mathsf{F}(k_j, (x_i, i))$ for all i. The simulator outputs $\pi = (j, \mathsf{OWF.InHom}(\mathsf{crs}_f, z_1, \ldots, z_T))$.

Take any Boolean circuit $C\colon \{0,1\}^n \times \{0,1\}^v \to \{0,1\}$, batch size T, and statements $x_1, \ldots, x_T$ and witnesses $w_1, \ldots, w_T$ such that $C(x_i, w_i) = 1$ for all $i \in [T]$. First, observe that the common reference string crs output by $\mathcal{S}_0(1^\lambda, T, C)$ is distributed identically to $\mathsf{Setup}(1^\lambda, T, C)$. It now suffices to consider the proof. By construction, the proof $\pi = (j, z)$ output by $\mathsf{P}(\mathsf{crs}, (x_1, \ldots, x_T), (w_1, \ldots, w_T))$ is obtained by evaluating $\mathsf{ObfGenSol}$ on inputs (i, j, x_i, w_i). By correctness of $i\mathcal{O}$ and the definition of GenSol and P, this means that j is the smallest value in $[T+1]$ such that $j \neq \mathsf{F}(k_{\mathsf{sel}}, (x_i, i))$ for all $i \in [T]$ and that $z_i = \mathsf{F}(k_j, (x_i, i))$ for all i. P then computes $z = \mathsf{OWF.InHom}(\mathsf{crs}_f, z_1, \ldots, z_T)$. Thus the proof output by $\mathcal{S}_1(\mathsf{st}, (x_1, \ldots, x_T))$ is distributed identically to π.

5 Re-randomizable PRGs

In this section, we introduce the notion of a re-randomizable PRG, which is one of the main building blocks we use in our construction of an adaptive fully succinct BARG in Sect. 6. Then, in Sect. 5.1, we show how to construct a re-randomizable PRG from DDH.

Definition 11 (Re-randomizable PRG). *A re-randomizable* PRG *is a tuple of efficient algorithms* $\Pi_{\mathsf{RPRG}} = (\mathsf{Setup}, \mathsf{Eval}, \mathsf{Rerandomize})$ *with the following syntax:*

- $\mathsf{Setup}(1^\lambda, 1^m) \to \mathsf{crs}$: *On input security parameter* 1^λ *and re-randomization parameter* 1^m*, the setup algorithm outputs a common reference string* crs*. We assume that the* crs *contains an implicit description of the input and output lengths of* $\mathsf{Eval}(\mathsf{crs}, \cdot)$*.*
- $\mathsf{Eval}(\mathsf{crs}, z) \to y$: *On input common reference string* crs *and seed* z*, the deterministic evaluation algorithm outputs an instance* y*.*
- $\mathsf{Rerandomize}(\mathsf{crs}, y) \to y'$: *On input common reference string* crs *and instance* y*, the randomization algorithm outputs a new instance* y'*.*

We require that Π_{RPRG} *satisfy the following properties:*

- ***Expansion.*** *For all* $\lambda, m \in \mathbb{N}$ *and all* crs *in the support of* $\mathsf{Setup}(1^\lambda, 1^m)$*, it holds that the output length of* $\mathsf{Eval}(\mathsf{crs}, \cdot)$ *is at least* λ *bits longer than the input length (i.e.,* $|y| \geq \lambda + |z|$*).*
- ***Pseudorandomness.*** *For a security parameter* λ*, a re-randomization parameter* m*, and a bit* $b \in \{0,1\}$*, we define the pseudorandomness security game between an adversary* $\mathcal{A}$ *and a challenger as follows:*
  - *The challenger samples* $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^m)$*.*
  - *If* $b = 0$*, the challenger samples* z *uniformly and computes* $y \leftarrow \mathsf{Eval}(\mathsf{crs}, z)$*. If* $b = 1$*, the challenger samples* y *uniformly.*
  - *The challenger then sends* (crs, y) *to* $\mathcal{A}$*.*
  - $\mathcal{A}$ *outputs a bit* b'*, which is the output of the experiment.*

  We say that Π_{RPRG} *is* (t, ε)*-pseudorandom if for all polynomials* $m = m(\lambda)$ *and all adversaries* $\mathcal{A}$ *running in time at most* $t(\lambda) \cdot \mathsf{poly}(\lambda)$*, there exists* $\lambda_{\mathcal{A}} \in \mathbb{N}$ *such that for all security parameters* $\lambda \geq \lambda_{\mathcal{A}}$*, it holds that*

$$\mathsf{PRGAdv}_{\mathcal{A}}(\lambda) := |\Pr[b' = 1 : b = 0] - \Pr[b' = 1 : b = 1]| \leq \varepsilon(\lambda)$$

  in the pseudorandomness security game.
- ***Re-randomization correctness.*** *For all* crs *in the support of* $\mathsf{Setup}(1^\lambda, 1^m)$*, all* y*, and all* y' *in the support of* $\mathsf{Rerandomize}(\mathsf{crs}, y)$*, we have either that* y, y' *are both in* $\mathsf{Im}(\mathsf{Eval}(\mathsf{crs}, \cdot))$ *or both not in* $\mathsf{Im}(\mathsf{Eval}(\mathsf{crs}, \cdot))$*.*
- ***Re-randomization security.*** *For a security parameter* λ*, a re-randomization parameter* m*, and a bit* $b \in \{0,1\}$*, we define the re-randomization security game between an adversary* $\mathcal{A}$ *and a challenger as follows:*

- *The challenger samples* $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^m)$ *and* $y_{\mathsf{base}} \leftarrow \mathsf{Eval}(\mathsf{crs}, z_{\mathsf{base}})$ *for uniform* z_{base}.
- *If* $b = 0$, *the challenger samples* $y^* \leftarrow \mathsf{Eval}(\mathsf{crs}, z^*)$ *for uniform* z^*. *If* $b = 1$, *the challenger samples* $y^* \leftarrow \mathsf{Rerandomize}(\mathsf{crs}, y_{\mathsf{base}})$.
- *The challenger then sends* $(\mathsf{crs}, y_{\mathsf{base}}, y^*)$ *to* $\mathcal{A}$.
- $\mathcal{A}$ *outputs a bit* b', *which is the output of the experiment.*

We say that Π_{RPRG} *satisfies* (t, ε)*-re-randomization security if for all polynomials* $m = m(\lambda)$ *and all adversaries* $\mathcal{A}$ *running in time at most* $t(\lambda) \cdot \mathsf{poly}(\lambda)$, *there exists* $\lambda_{\mathcal{A}} \in \mathbb{N}$ *such that for all security parameters* $\lambda \geq \lambda_{\mathcal{A}}$, *it holds that*

$$\mathsf{RerandAdv}_{\mathcal{A}}(\lambda) := |\Pr[b' = 1 : b = 0] - \Pr[b' = 1 : b = 1]| \leq \varepsilon(\lambda)$$

in the re-randomization security game.

– ***Succinctness.*** *There exists a polynomial* p *such that for all* $\lambda, m \in \mathbb{N}$ *and all* crs *in the support of* $\mathsf{Setup}(1^\lambda, 1^m)$, *it holds that the input length of* $\mathsf{Eval}(\mathsf{crs}, \cdot)$ *is* $|z| \leq p(\lambda + \log m)$.

5.1 Constructing Re-randomizable PRGs

In this section, we show how to construct a re-randomizable PRG from the decisional Diffie-Hellman assumption.

Construction 3 (Re-randomizable PRG). *Let* GroupGen *be a prime-order group generator. We construct a re-randomizable* PRG $\Pi_{\mathsf{RPRG}} = (\mathsf{Setup}, \mathsf{Eval}, \mathsf{Rerandomize})$ *as follows:*

– $\mathsf{Setup}(1^\lambda, 1^m)$: *On input security parameter* 1^λ *and rerandomization parameter* 1^m, *the setup algorithm samples* $\mathcal{G} = (\mathbb{G}, p, g) \leftarrow \mathsf{GroupGen}(1^\lambda)$, *samples* $x \xleftarrow{R} \mathbb{Z}_p^*$ *and outputs* $\mathsf{crs} = (\mathbb{G}, p, g, h = g^x)$.
– $\mathsf{Eval}(\mathsf{crs}, z)$: *On input common reference string* $\mathsf{crs} = (\mathbb{G}, p, g, h)$ *and seed* z, *the evaluation algorithm parses* $z \in \mathbb{Z}_p^*$ *and outputs* (g^z, h^z).
– $\mathsf{Rerandomize}(\mathsf{crs}, y)$: *On input common reference string* crs *and instance* $y = (y_1, y_2)$, *the rerandomization algorithm samples* $r \xleftarrow{R} \mathbb{Z}_p^*$ *and outputs* (y_1^r, y_2^r).

Theorem 7 (Expansion). *Construction 3 satisfes expansion.*

Proof. For $(\mathbb{G}, p, g) \leftarrow \mathsf{GroupGen}(1^\lambda)$, we have that p is a λ-bit prime, so a seed $z \in \mathbb{Z}_p^*$ has length λ, while an instance $y = (g^z, h^z)$ consists of two group elements and thus has length at least 2λ.

Theorem 8 (Pseudorandomness). *Suppose the decisional Diffie-Hellman assumption holds with respect to* GroupGen. *Then Construction 3 satisfies pseudorandomness.*

Proof. Let $\mathcal{A}$ be an efficient adversary for the pseudorandomness game against Construction 3. We use $\mathcal{A}$ to construct an adversary $\mathcal{B}$ for the DDH problem.

Algorithm $\mathcal{B}$

Inputs: $((\mathbb{G}, p, g), g^\alpha, g^\beta, g^\gamma)$ from challenger
1. Let $\mathsf{crs} = (\mathbb{G}, p, g, h)$ where $h = g^\alpha$ and $y = (g^\beta, g^\gamma)$.
2. Run $b' \leftarrow \mathcal{A}(\mathsf{crs}, y)$.
3. Send b' to challenger.

Note that if $b = 0$ (i.e., $\gamma = \alpha\beta$ for uniform α, β), then $g^\gamma = g^{\alpha\beta} = h^\beta$ so $y = (g^\beta, h^\beta) = \mathsf{Eval}(\mathsf{crs}, \beta)$. If $b = 1$ (i.e., γ is independently uniform), then $y = (g^\beta, g^\gamma)$ is uniform. Thus

$$\mathsf{PRGAdv}_{\mathcal{A}}(\lambda) = \mathsf{DDHAdv}_{\mathcal{B}}(\lambda) \leq \mathsf{negl}(\lambda).\square$$

Theorem 9 (Re-randomization Correctness). *Construction 3 satisfies re-randomization correctness.*

Proof. Take any $\lambda, m \in \mathbb{N}$, any common reference string $\mathsf{crs} = (\mathbb{G}, p, g, h = g^x)$ in the support of $\mathsf{Setup}(1^\lambda, 1^m)$, any y, and any y' in the support of $\mathsf{Rerandomize}(\mathsf{crs}, y)$.

- If $y = (y_1, y_2) = \mathsf{Eval}(\mathsf{crs}, z)$ for some $z \in \mathbb{Z}_p^*$, then $y_1 = g^z$ and $y_2 = h^z$. We have that for some $r \in \mathbb{Z}_p^*$, $y' = (y_1^r, y_2^r) = (g^{rz}, h^{rz}) = \mathsf{Eval}(\mathsf{crs}, rz)$ where $rz \in \mathbb{Z}_p^*$.
- If $y' = (y_1', y_2') = \mathsf{Eval}(\mathsf{crs}, z)$ for some $z \in \mathbb{Z}_p^*$, then $y_1' = g^z$ and $y_2' = h^z$. We have that for some $r \in \mathbb{Z}_p^*$, $y = ({y_1'}^{r^{-1}}, {y_2'}^{r^{-1}}) = (g^{r^{-1}z}, h^{r^{-1}z}) = \mathsf{Eval}(\mathsf{crs}, r^{-1}z)$ where $r^{-1}z \in \mathbb{Z}_p^*$.

Thus, either y, y' are both in $\mathsf{Im}(\mathsf{Eval}(\mathsf{crs}, \cdot))$ or both not in $\mathsf{Im}(\mathsf{Eval}(\mathsf{crs}, \cdot))$.

Theorem 10 (Re-randomization Security). *Construction 3 satisfies perfect re-randomizable security. Namely, for all polynomials $m = m(\lambda)$ and all adversaries $\mathcal{A}$, $\mathsf{RerandAdv}_{\mathcal{A}}(\lambda) = 0$.*

Proof. Take any polynomial $m = m(\lambda)$. Let $\mathsf{crs} = (\mathbb{G}, p, g, h = g^x) \leftarrow \mathsf{Setup}(1^\lambda, 1^m)$. By construction of Eval, we have that a fresh instance $y = (g^{z^*}, h^{z^*})$ is uniformly distributed over the set $\{(g^z, h^z) \mid z \in \mathbb{Z}_p^*\}$. By construction of $\mathsf{Rerandomize}$, we have that a rerandomized instance $y' = (y_1^r, y_2^r) = (g^{rz}, h^{rz})$ is still uniformly distributed over the set $\{(g^z, h^z) \mid z \in \mathbb{Z}_p^*\}$, since rz is uniformly distributed over $\mathbb{Z}_p^*$.

6 SNARG for Batch NP from Re-randomizable PRGs

Construction 4. (Adaptive Batch Argument for NP). *Let λ be a security parameter. We construct a BARG scheme that supports NP languages with an arbitrary polynomial number $T = T(\lambda) < 2^\lambda$ of instances of length $n = n(\lambda)$. Our construction will leverage sub-exponential hardness of the below primitives (except for pseudorandomness of Π_{RPRG}). Our construction relies on the following primitives:*

- *Let $i\mathcal{O}$ be an indistinguishability obfuscator for Boolean circuits.*
- *Let $\Pi_{\mathsf{SEH}} = (\mathsf{H.Setup}, \mathsf{H.Hash}, \mathsf{H.Open}, \mathsf{H.Verify}, \mathsf{H.Extract})$ be a somewhere extractable hash family.*
- *Let $\Pi_{\mathsf{PPRF}} = (\mathsf{F.Setup}, \mathsf{F.Eval}, \mathsf{F.Puncture})$ be a puncturable PRF. For a key k and an input x, we will write $\mathsf{F}(k, x)$ to denote $\mathsf{F.Eval}(k, x)$.*
- *Let $\Pi_{\mathsf{RPRG}} = (\mathsf{G.Setup}, \mathsf{G.Eval}, \mathsf{G.Rerandomize})$ be a re-randomizable* PRG*. For parameters $\mathsf{crs}_{\mathsf{G}}$ and a seed z, we write $\mathsf{G}(\mathsf{crs}_{\mathsf{G}}, z)$ to denote $\mathsf{G.Eval}(\mathsf{crs}_{\mathsf{G}}, z)$.*

We will describe how to define the polynomials $\lambda_{\mathsf{SEH}}, \lambda_{\mathsf{obf}}, \lambda_{\mathsf{PRF}}$, and m in the security analysis. We construct a fully succinct non-interactive batch argument $\Pi_{\mathsf{BARG}} = (\mathsf{Gen}, \mathsf{P}, \mathsf{V})$ for NP as follows:

- $\mathsf{Setup}(1^\lambda, T, C) \to \mathsf{crs}$*: On input security parameter 1^λ, batch size T, and Boolean circuit $C\colon \{0,1\}^n \times \{0,1\}^v \to \{0,1\}$, the setup algorithm does the following:*
  - *Sample* PRG *parameters* $\mathsf{crs}_{\mathsf{G}} \leftarrow \mathsf{G.Setup}(1^\lambda, 1^m)$.
  - *Sample an* SEH *hash key* $\mathsf{hk} \leftarrow \mathsf{H.Setup}(1^{\lambda_{\mathsf{SEH}}}, 1^n)$.
  - *Let $t = \log(T+1)$. Let n' be the output length of* $\mathsf{H.Hash}(\mathsf{hk}, \cdot)$*. Let ρ be the input length of* $\mathsf{G}(\mathsf{crs}, \cdot)$*. Let τ be the number of bits of randomness the* $\mathsf{F.Setup}(1^{\lambda_{\mathsf{PRF}}}, 1^{n'+t}, 1^\rho)$ *algorithm takes.*
  - *Sample a "selector"* PPRF *key* $k_{\mathsf{sel}} \leftarrow \mathsf{F.Setup}(1^{\lambda_{\mathsf{PRF}}}, 1^{n+t}, 1^t)$.
  - *Sample a "key generator"* PPRF *key* $k \leftarrow \mathsf{F.Setup}(1^{\lambda_{\mathsf{PRF}}}, 1^t, 1^\tau)$.
  - *Define the* Verify *program with the* PRG *parameters* $\mathsf{crs}_{\mathsf{G}}$ *and* PPRF *key k hardcoded:*

    $\mathsf{Verify}[\mathsf{crs}_{\mathsf{G}}, k]$

    Inputs: *index i, selection symbol j, hash value* dig*, proof z_i*
    1. *Compute* $k_j \leftarrow \mathsf{F.Setup}(1^{\lambda_{\mathsf{PRF}}}, 1^{n'+t}, 1^\rho; \mathsf{F}(k, j))$.
    2. *If* $\mathsf{G}(\mathsf{crs}_{\mathsf{G}}, z_i) = \mathsf{G}(\mathsf{crs}_{\mathsf{G}}, \mathsf{F}(k_j, (\mathsf{dig}, i)))$*, output 1.*
    3. *Output 0.*

  - *Define the* AggProof *program (which has the code for* Verify *replicated inside) with the circuit C,* PRG *parameters* $\mathsf{crs}_{\mathsf{G}}$*,* SEH *hash key* hk*, and* PPRF *keys* k_{sel}, k *hardcoded:*

    $\mathsf{AggProof}[C, \mathsf{crs}_{\mathsf{G}}, \mathsf{hk}, k_{\mathsf{sel}}, k]$

    Inputs: *index i, selection symbol j, hash value* dig*, statement x_i, witness w_i, opening σ_i, prior proof z_{i-1}*
    1. *If* $C(x_i, w_i) = 0$*, output* $\perp$.
    2. *If* $\mathsf{H.Verify}(\mathsf{hk}, \mathsf{dig}, x_i, i, \sigma_i) = 0$*, output* $\perp$.
    3. *If* $j = \mathsf{F}(k_{\mathsf{sel}}, (x_i, i))$*, output* $\perp$.
    4. *If* $i \neq 1$ *and* $\mathsf{Verify}(i-1, j, \mathsf{dig}, z_{i-1}) = 0$*, output* $\perp$.
    5. *Compute* $k_j \leftarrow \mathsf{F.Setup}(1^{\lambda_{\mathsf{PRF}}}, 1^{n'+t}, 1^\rho; \mathsf{F}(k, j))$.
    6. *Output* $z_i = \mathsf{F}(k_j, (\mathsf{dig}, i))$.

- *Let* $s = s(\lambda, n, |C|)$ *be the maximum size of the* $\mathsf{AggProof}$ *and* Verify *programs as well as those appearing in the security analysis.*
- *Construct the obfuscated programs*

$$\mathsf{ObfAggProof} \leftarrow i\mathcal{O}(1^{\lambda_{\mathsf{obf}}}, 1^s, \mathsf{AggProof}[C, \mathsf{crs}_{\mathsf{G}}, \mathsf{hk}, k_{\mathsf{sel}}, k])$$

and

$$\mathsf{ObfVerify} \leftarrow i\mathcal{O}(1^{\lambda_{\mathsf{obf}}}, 1^s, \mathsf{Verify}[\mathsf{crs}_{\mathsf{G}}, k]).$$

- *Output* $\mathsf{crs} = (\mathsf{hk}, \mathsf{crs}_{\mathsf{G}}, \mathsf{ObfAggProof}, \mathsf{ObfVerify})$.

– $\mathsf{P}(\mathsf{crs}, (x_1, \ldots, x_T), (w_1, \ldots, w_T)) \to \pi = (j, z)$*:*
- *Compute* $\mathsf{dig} \leftarrow \mathsf{H.Hash}(\mathsf{hk}, (x_1, \ldots, x_T))$.
- *Set* $i = 1, j = 1$ *and* $z_0 = \emptyset$.
- *While* $i \leq T$:
  - *Compute* $\sigma_i \leftarrow \mathsf{H.Open}(\mathsf{hk}, (x_1, \ldots, x_T), i)$.
  - *Compute* $z_i \leftarrow \mathsf{ObfAggProof}(i, j, \mathsf{dig}, x_i, w_i, \sigma_i, z_{i-1})$.
  - *If* $z_i = \perp$, *set* $i = 1, j = j + 1$.
  - *Else, set* $i = i + 1$.
- *Output* (j, z_T).

– $\mathsf{V}(\mathsf{crs}, (x_1, \ldots, x_T), \pi = (j, z_T)) \to 0/1$*:*
- *Compute* $\mathsf{dig} \leftarrow \mathsf{H.Hash}(\mathsf{hk}, (x_1, \ldots, x_T))$.
- *Output* $\mathsf{ObfVerify}(T, j, \mathsf{dig}, z_T)$.

Theorem 11. (Completeness). *Suppose* $i\mathcal{O}$ *is correct and* Π_{SEH} *satisfies opening completness. Then Construction 4 is complete.*

Proof. Take any security parameter $\lambda \in \mathbb{N}$, any Boolean circuit $C\colon \{0,1\}^n \times \{0,1\}^v \to \{0,1\}$, any $T \leq 2^\lambda$, and any statements $(x_1, \ldots, x_T)$ and witnesses $(w_1, \ldots, w_T)$ such that $C(x_i, w_i) = 1$ for all $i \in [T]$. Let

$$\mathsf{crs} = (\mathsf{hk}, \mathsf{crs}_{\mathsf{G}}, \mathsf{ObfAggProof}, \mathsf{ObfVerify}) \leftarrow \mathsf{Setup}(1^\lambda, C, T)$$

and $\pi = (j, z_T) \leftarrow \mathsf{P}(\mathsf{crs}, (x_1, \ldots, x_T), (w_1, \ldots, w_T))$. Consider the output of $\mathsf{V}(\mathsf{crs}, (x_1, \ldots, x_T), \pi)$:

– By construction, $\mathsf{ObfAggProof}$ is an obfuscation of the aggregation program $\mathsf{AggProof}[C, \mathsf{crs}_{\mathsf{G}}, \mathsf{hk}, k_{\mathsf{sel}}, k]$, where

$$\begin{aligned} \mathsf{crs}_{\mathsf{G}} &\leftarrow \mathsf{G.Setup}(1^\lambda, 1^m) \\ \mathsf{hk} &\leftarrow \mathsf{H.Setup}(1^{\lambda_{\mathsf{SEH}}}, 1^n) \\ k_{\mathsf{sel}} &\leftarrow \mathsf{F.Setup}(1^{\lambda_{\mathsf{PRF}}}, 1^{n+t}, 1^t) \\ k &\leftarrow \mathsf{F.Setup}(1^{\lambda_{\mathsf{PRF}}}, 1^t, 1^\tau). \end{aligned}$$

Algorithm P obtains (j, z_T) by computing $\mathsf{dig} = \mathsf{H.Hash}(\mathsf{hk}, (x_1, \ldots, x_T))$ (along with openings σ_i to x_i for all i) and evaluating $\mathsf{ObfAggProof}$ on inputs $(i, j, \mathsf{dig}, x_i, w_i, \sigma_i, z_{i-1})$ to obtain intermediate proofs z_{i-1}. By correctness of $i\mathcal{O}$ and the definition of GenSol, this means that $z_i = \mathsf{F}(k_j, (\mathsf{dig}, i))$ if $i = 1$ or $\mathsf{Verify}(i - 1, j, \mathsf{dig}, z_{i-1}) = 1$.

- We then see by induction that $z_i = \mathsf{F}(k_j, (\mathsf{dig}, i))$ for all $i \in [T]$, since $\mathsf{Verify}(i-1, j, \mathsf{dig}, z_{i-1})$ outputs 1 if $\mathsf{G}(\mathsf{crs}_\mathsf{G}, z_{i-1}) = \mathsf{G}(\mathsf{crs}_\mathsf{G}, \mathsf{F}(k_j, (\mathsf{dig}, i-1))$, which follows from the fact that $z_{i-1} = \mathsf{F}(k_j, (\mathsf{dig}, i-1))$.
- By construction, $\mathsf{ObfVerify}$ is an obfuscation of the program $\mathsf{Verify}[\mathsf{crs}_\mathsf{G}, k]$. The verification algorithn V computes $\mathsf{dig} = \mathsf{H.Hash}(\mathsf{hk}, (x_1, \ldots, x_T))$ and outputs $b \leftarrow \mathsf{ObfVerify}(T, j, \mathsf{dig}, z_T)$. By correctness of $i\mathcal{O}$ and the definition of Verify, we know $b = 1$ if $\mathsf{G}(\mathsf{crs}_\mathsf{G}, z_T) = \mathsf{G}(\mathsf{crs}_\mathsf{G}, \mathsf{F}(k_j, (\mathsf{dig}, T))$, which follows from the fact that $z_T = \mathsf{F}(k_j, (\mathsf{dig}, T))$.

Theorem 12 (Succinctness). *Suppose Π_{SEH} and Π_{RPRG} are succinct. Then Construction 4 is succinct.*

Proof. A proof (j, z_T) in Construction 4 consists of a selection symbol $j \in [T+1]$ and PRG seed z_T. Since Π_{RPRG} is succinct, there is a fixed polynomial p such that $|z| \leq p(\lambda + \log m)$. In Construction 4, $m(\lambda, n')$ is a fixed polynomial in the security parameter λ and n', which is the output length of $\mathsf{H.Hash}(\mathsf{hk}, \cdot)$ for $\mathsf{hk} \leftarrow \mathsf{H.Setup}(1^{\lambda_{\mathsf{SEH}}}, 1^n)$ where λ_{SEH} is a fixed polynomial in the witness length v and λ. By succinctness of Π_{SEH}, we have that n' is a fixed polynomial in λ, n, and v. The statement length and witness length are always upper-bounded by the circuit size, so it follows that $|\pi| \leq \mathsf{poly}(\lambda + \log |C|) + \log T$.

Theorem 13. (Adaptive Soundness). *Suppose $i\mathcal{O}$ is $(1, 2^{-\lambda_{\mathsf{obf}}^{\varepsilon_{\mathsf{obf}}}})$-secure, Π_{SEH} satisfies statistical binding and $(2^{\lambda_{\mathsf{SEH}}^{\varepsilon_{\mathsf{SEH}}}}, \mathsf{negl}(\lambda_{\mathsf{SEH}}))$-index hiding security, Π_{PPRF} satisfies punctured correctness and $(1, 2^{-\lambda_{\mathsf{PRF}}^{\varepsilon_{\mathsf{PRF}}}})$-puncturing security, and Π_{RPRG} expands by λ bits and satisfies $(1, \mathsf{negl}(\lambda))$-pseudorandomness and $(1, 2^{-m^{\varepsilon_m}})$-rerandomization security for constants $(\varepsilon_{\mathsf{SEH}}, \varepsilon_{\mathsf{obf}}, \varepsilon_{\mathsf{PRF}}, \varepsilon_m) \in (0,1)$ and security parameters $\lambda_{\mathsf{SEH}} = (v + \omega(\log \lambda))^{1/\varepsilon_{\mathsf{SEH}}}, \lambda_{\mathsf{obf}} = (\lambda + n')^{1/\varepsilon_{\mathsf{obf}}}, \lambda_{\mathsf{PRF}} = (\lambda + n')^{1/\varepsilon_{\mathsf{PRF}}}, m = (\lambda + n')^{1/\varepsilon_m}$ where n' is the length of $\mathsf{H.Hash}(\mathsf{H.Setup}(1^{\lambda_{\mathsf{SEH}}}, 1^n), \cdot)$. Then Construction 4 satisfies adaptive soundness.*

Proof. We refer to the full version for a proof of adaptive soundness.

Theorem 14 (Perfect Zero-Knowledge). *Suppose $i\mathcal{O}$ is correct. Then Construction 4 satisfies perfect zero-knowledge.*

Proof. We construct the simulator as follows:

- $\mathcal{S}_0(1^\lambda, T, C)$: On input security parameter 1^λ, batch size T, and Boolean circuit $C\colon \{0,1\}^n \times \{0,1\}^v \to \{0,1\}$, the simulator samples the common reference string $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, T, C)$ exactly as in the real scheme. Let $\mathsf{hk}, \mathsf{crs}_\mathsf{G}, k_{\mathsf{sel}}, k$ be the underlying hash key, PRG parameters and PPRF keys sampled in Setup. The simulator outputs the crs along with the state $\mathsf{st} = (\mathsf{hk}, \mathsf{crs}_\mathsf{G}, k_{\mathsf{sel}}, k)$.
- $\mathcal{S}_1(\mathsf{st}, (x_1, \ldots, x_T))$: On input the state $\mathsf{st} = (\mathsf{hk}, \mathsf{crs}_\mathsf{G}, k_{\mathsf{sel}}, k)$ and statements $(x_1, \ldots, x_T)$, the simulator computes $j_i \leftarrow \mathsf{F}(k_{\mathsf{sel}}, (x_i, i))$ and selects the smallest $j \in [T+1]$ such that $j \neq j_i$ for all $i \in [T]$. It then computes $k_j \leftarrow \mathsf{F.Setup}(1^{\lambda_{\mathsf{PRF}}}, 1^{n+t}, 1^\lambda; \mathsf{F}(k, j))$ and $z_T \leftarrow \mathsf{F}(k_j, (\mathsf{dig}, T))$. The simulator outputs $\pi = (j, z_T)$.

Take any Boolean circuit $C\colon \{0,1\}^n \times \{0,1\}^v \to \{0,1\}$, batch size T, and statements $x_1, \ldots, x_T$ and witnesses $w_1, \ldots, w_T$ such that $C(x_i, w_i) = 1$ for all $i \in [T]$. First, observe that the common reference string crs output by $\mathcal{S}_0(1^\lambda, T, C)$ is distributed identically to $\mathsf{Setup}(1^\lambda, T, C)$. It now suffices to consider the proof. By construction, the proof $\pi = (j, z_T)$ output by $\mathsf{P}(\mathsf{crs}, (x_1, \ldots, x_T), (w_1, \ldots, w_T))$ is obtained by evaluating $\mathsf{ObfGenSol}$ on inputs $(i, j, \mathsf{dig}, x_i, w_i, \sigma_i, z_{i+1})$. By correctness of $i\mathcal{O}$ and the definition of GenSol and P, this means that j is the smallest value in $[T+1]$ such that $j \neq \mathsf{F}(k_{\mathsf{sel}}, (x_i, i))$ for all $i \in [T]$ and that $z_T = \mathsf{F}(k_j, (\mathsf{dig}, T))$. Thus the proof $\pi = (j, z_T)$ output by $\mathcal{S}_1(\mathsf{st}, (x_1, \ldots, x_T))$ is distributed identically to $\mathsf{P}(\mathsf{crs}, (x_1, \ldots, x_T), (w_1, \ldots, w_T))$.

Acknowledgments. Brent Waters is supported by NSF CNS-1908611, CNS-2318701, and a Simons Investigator award. David J. Wu is supported by NSF CNS-2140975, CNS-2318701, a Microsoft Research Faculty Fellowship, and a Google Research Scholar award.

References

1. Albrecht, M.R., Cini, V., Lai, R.W.F., Malavolta, G., Thyagarajan, S.A.K.: Lattice-based SNARKs: publicly verifiable, preprocessing, and recursively composable: (extended abstract). In: Dodis, Y., Shrimpton, T. (eds.) CRYPTO 2022. LNCS, vol. 13508, Part II, pp. 102–132. Springer, Cham (2022)
2. Barak, B., et al.: On the (im)possibility of obfuscating programs. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 1–18. Springer, Heidelberg (2001)
3. Bitansky, N., et al.: The hunting of the SNARK. J. Cryptol. **30**(4), 989–1066 (2017)
4. Bitansky, N., Canetti, R., Chiesa, A., Tromer, E.: From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again. In: Goldwasser, S. (ed.) ITCS 2012, pp. 326–349. ACM (2012)
5. Bitansky, N., Canetti, R., Paneth, O., Rosen, A.: On the existence of extractable one-way functions. In: Shmoys, D.B. (ed.) 46th ACM STOC, pp. 505–514. ACM Press (2014)
6. Bitansky, N., Chiesa, A., Ishai, Y., Ostrovsky, R., Paneth, O.: Succinct non-interactive arguments via linear interactive proofs. In: Sahai, A. (ed.) TCC 2013. LNCS, vol. 7785, pp. 315–333. Springer, Heidelberg (2013)
7. Boneh, D., Ishai, Y., Sahai, A., Wu, D.J.: Lattice-based SNARGs and their application to more efficient obfuscation. In: Coron, J.-S., Nielsen, J.B. (ed.) EUROCRYPT 2017, Part III, LNCS, vol. 10212, pp. 247–277. Springer, Heidelberg (2017)
8. Boneh, D., Waters, B.: Constrained pseudorandom functions and their applications. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT 2013, LNCS, Part II, vol. 8270, pp. 280–300. Springer, Heidelberg (2013)
9. Boyle, E., Goldwasser, S., Ivan, I.: Functional signatures and pseudorandom functions. In: Krawczyk, H. (ed.) PKC 2014. LNCS, vol. 8383, pp. 501–519. Springer, Heidelberg (2014)
10. Brakerski, Z., Brodsky, M.F., Kalai, Y.T., Lombardi, A., Paneth, O.: SNARGs for monotone policy batch NP. In: Handschuh, H., Lysyanskaya, A. (eds.) CRYPTO 2023. LNCS, Part II, vol. 14082, pp. 252–283. Springer, Heidelberg (2023)
11. Chiesa, A., Tromer, E.: Proof-carrying data and hearsay arguments from signature cards. In: Yao, A.C.-C. (ed.) ICS 2010, pp. 310–331. Tsinghua University Press (2010)

12. Choudhuri, A.R., Garg, S., Jain, A., Jin, Z., Zhang, J.: Correlation intractability and SNARGs from sub-exponential DDH. In: Handschuh, H., Lysyanskaya, A. (eds.) CRYPTO 2023. LNCS, Part IV, vol. 14084, pp. 635–668. Springer, Heidelberg (2023)
13. Choudhuri, A.R., Jain, A., Jin, Z.: Non-interactive batch arguments for NP from standard assumptions. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021. LNCS, Part IV, vol. 12828, pp. 394–423. Springer, Heidelberg (2021), Virtual Event
14. Choudhuri, A.R., Jain, A., Jin, Z.: SNARGs for $\mathcal{P}$ from LWE. In: 62nd FOCS, pp. 68–79. IEEE Computer Society Press (2022)
15. Cini, V., Lai, R.W.F., Malavolta, G.: Lattice-based succinct arguments from vanishing polynomials - (extended abstract). In: Handschuh, H., Lysyanskaya, A. (eds.) CRYPTO 2023. LNCS, Part II, vol. 14082, pp. 72–105. Springer, Heidelberg (2023)
16. Damgård, I., Faust, S., Hazay, C.: Secure two-party computation with low communication. In: Cramer, R. (ed.) TCC 2012. LNCS, vol. 7194, pp. 54–74. Springer, Heidelberg (2012)
17. Devadas, L., Goyal, R., Kalai, Y., Vaikuntanathan, V.: Rate-1 non-interactive arguments for batch-NP and applications. In: 63rd FOCS, pp. 1057–1068. IEEE Computer Society Press (2022)
18. Garg, R., Sheridan, K., Waters, B., David, J.W.: Fully succinct batch arguments for NP from indistinguishability obfuscation. In: Kiltz, E., Vaikuntanathan, V. (eds.) TCC 2022. LNCS, Part I, vol. 13747, pp. 526–555. Springer, Heidelberg (2022)
19. Gennaro, R., Gentry, C., Parno, B., Raykova, M.: Quadratic span programs and succinct NIZKs without PCPs. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 626–645. Springer, Heidelberg (2013)
20. Gentry, C., Wichs, D.: Separating succinct non-interactive arguments from all falsifiable assumptions. In: Fortnow, L., Vadhan, S.P. (eds.) 43rd ACM STOC, pp. 99–108. ACM Press (2011)
21. Groth, J.: Short pairing-based non-interactive zero-knowledge arguments. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 321–340. Springer, Heidelberg (2010)
22. Hubacek, P., Wichs, D.: On the communication complexity of secure function evaluation with long output. In: Roughgarden, T. (ed.) ITCS 2015, pp. 163–172. ACM (2015)
23. Hulett, J., Jawale, R., Khurana, D., Srinivasan, A.: SNARGs for P from sub-exponential DDH and QR. In: Dunkelman, O., Dziembowski, S. (eds.) EUROCRYPT 2022. LNCS, Part II, vol. 13276, pp. 520–549. Springer, Heidelberg (2022)
24. Jain, A., Lin, H., Sahai, A.: Indistinguishability obfuscation from well-founded assumptions. In: Khuller, S., Williams, V.V. (eds.) 53rd ACM STOC, pp. 60–73. ACM Press (2021)
25. Jain, A., Lin, H., Sahai, A.: Indistinguishability obfuscation from LPN over $\mathbb{F}_p$, DLIN, and PRGs in NC^0. In: Dunkelman, O., Dziembowski, S. (eds.) EUROCRYPT 2022. LNCS, Part I, vol. 13275, pages 670–699. Springer, Heidelberg (2022)
26. Kalai, Y., Lombardi, A., Vaikuntanathan, V., Wichs, D.: Boosting batch arguments and RAM delegation. In: Saha, B., Servedio, R.A., (eds.) 55th ACM STOC, pp. 1545–1552. ACM Press (2023)
27. Kalai, Y.T., Lombardi, A., Vaikuntanathan, V.: SNARGs and PPAD hardness from the decisional Diffie-Hellman assumption. In: Hazay, C., Stam, M. (eds.) EUROCRYPT 2023. LNCS, Part II, vol. 14005, pp. 470–498. Springer, Heidelberg (2023)
28. Kalai, Y.T., Paneth, O., Yang, L.: How to delegate computations publicly. In: Charikar, M., Cohen, E. (eds.) 51st ACM STOC, pp. 1115–1124. ACM Press (2019)

29. Kalai, Y.T., Vaikuntanathan, V., Zhang, R.Y.: Somewhere statistical soundness, post-quantum security, and SNARGs. In: Nissim, K., Waters, B. (eds.) TCC 2021. LNCS, Part I, vol. 13042, pp. 330–368. Springer, Heidelberg (2021)
30. Kiayias, A., Papadopoulos, S., Triandopoulos, N., Zacharias, T.: Delegatable pseudorandom functions and applications. In: Sadeghi, A.-R., Gligor, V.D., Yung, M. (eds.) ACM CCS 2013, pp. 669–684. ACM Press (2013)
31. Kilian, J.: A note on efficient zero-knowledge proofs and arguments (extended abstract). In: 4th ACM STOC, pp. 723–732. ACM Press (1992)
32. Lipmaa, H.: Succinct non-interactive zero knowledge arguments from span programs and linear error-correcting codes. In: Sako, K., Sarkar, P. (eds.) ASIACRYPT 2013. LNCS, Part I, vol. 8269, pp. 41–60. Springer, Heidelberg (2013)
33. Mathialagan, S., Peters, S., Vaikuntanathan, V.: Adaptively sound zero-knowledge SNARKs for UP. In: CRYPTO (2024)
34. Micali, S.: CS proofs (extended abstracts). In: 35th FOCS, pp. 436–453. IEEE Computer Society Press (1994)
35. Nassar, S., Waters, B., Wu, D.J.: Monotone policy BARGs from BARGs and additively homomorphic encryption. In: TCC (2024)
36. Paneth, O., Pass, R.: Incrementally verifiable computation via rate-1 batch arguments. In: 63rd FOCS, pp. 1045–1056. IEEE Computer Society Press (2022)
37. Sahai, A., Waters, B:. How to use indistinguishability obfuscation: deniable encryption, and more. In: Shmoys, D.B. (ed.) 46th ACM STOC, pp. 475–484. ACM Press (2014)
38. Valiant, P.: Incrementally verifiable computation or proofs of knowledge imply time/space efficiency. In: Canetti, R. (ed.) TCC 2008. LNCS, vol. 4948, pp. 1–18. Springer, Heidelberg (2008)
39. Waters, B., David, J.W.: Batch arguments for NP and more from standard bilinear group assumptions. In: Dodis, Y., Shrimpton, T. (eds.) CRYPTO 2022. LNCS, Part II, vol. 13508, pp. 433–463. Springer, Heidelberg (2022)
40. Waters, B., Wu, D.J.: Adaptively-sound succinct arguments for NP from indistinguishability obfuscation. In: STOC (2024)
41. Waters, B., Wu, D.J.: A pure indistinguishability obfuscation approach to adaptively-sound SNARGs for NP. Cryptology ePrint Archive, Paper 2024/933 (2024)
42. Waters, B., Zhandry, M.: Adaptive security in SNARGs via iO and lossy functions. In: CRYPTO (2024)

Doubly-Efficient Batch Verification in Statistical Zero-Knowledge

Or Keret[1(✉)], Ron D. Rothblum[1], and Prashant Nalini Vasudevan[2]

[1] Technion, Haifa, Israel
{or.keret,rothblum}@cs.technion.ac.il
[2] National University of Singapore, Singapore, Singapore
prashant@comp.nus.edu.sg

Abstract. A sequence of recent works, concluding with Mu *et al.* (Eurocrypt, 2024) has shown that every problem Π admitting a non-interactive statistical zero-knowledge proof (NISZK) has an efficient zero-knowledge *batch verification* protocol. Namely, an NISZK protocol for proving that $x_1, \ldots, x_k \in \Pi$ with communication that only scales poly-logarithmically with k. A caveat of this line of work is that the prover runs in exponential-time, whereas for NP problems it is natural to hope to obtain a *doubly-efficient proof* – that is, a prover that runs in polynomial-time given the k NP witnesses.

In this work we show that every problem in NISZK $\cap$ UP has a *doubly-efficient* interactive statistical zero-knowledge proof with communication $\mathrm{poly}(n, \log(k))$ and $\mathrm{poly}(\log(k), \log(n))$ rounds. The prover runs in time $\mathrm{poly}(n, k)$ given access to the k UP witnesses. Here n denotes the length of each individual input, and UP is the subclass of NP relations in which YES instances have unique witnesses.

This result yields doubly-efficient statistical zero-knowledge batch verification protocols for a variety of concrete and central cryptographic problems from the literature.

1 Introduction

Suppose that a server, holding a long list of RSA public-keys $N_1, \ldots, N_k$ together with their factorizations, wants to convince an auditor that the public-keys are well formed. That is, that each N_i is a product of exactly two distinct primes. One option is to reveal all of the factorizations but this option is simultaneously extremely bad from a security perspective (as it reveals the clients' secret keys) and is highly inefficient if k is very large.

Efficient zero-knowledge proofs are protocols that allow short and easy-to-verify proofs of complex statements, in such a way that reveals nothing beyond the fact that the statement being proved is true. To solve the above problem one might employ a general purpose zero-knowledge proof from the literature. However, such proofs rely on unproven assumptions and provide only a computational soundness guarantee.

E. Boyle and M. Mahmoody (Eds.): TCC 2024, LNCS 15365, pp. 371–398, 2025.
https://doi.org/10.1007/978-3-031-78017-2_13

In this work, we are interested in *statistical zero-knowledge proofs* for batch verification. These are zero-knowledge proofs, in which both soundness and zero-knowledge hold information theoretically (and against unbounded adversaries) such that the communication required to prove the correctness of k statements scales sub-linearly (ideally, merely poly-logarithmically) with k. The information theoretic nature of such proofs also means that typically they do not involve expensive cryptographic operations, and can be highly efficient.

Returning to the above question on batch verification of prime products, a protocol due to Gennaro *et al.* [GMR98] gives a (non-interactive) statistical zero-knowledge proof that N is a product of two distinct primes. Running this protocol separately on all of the alleged RSA moduli $N_1, \ldots, N_k$, we could indeed prove in statistical zero-knowledge that all of them are prime products, alas with communication that scales linearly with k.

An alternative approach from a recent sequence of works [KRV21, KRR+20, MNRV24] has shown that *every* problem that has a non-interactive statistical zero-knowledge proof also has such a zero-knowledge proof for batch verification, in which the communication only scales poly-logarithmically with k. Unfortunately, these protocols have a prover that runs in *exponential-time*, whereas, for our example above, we would like for our server, who knows all of the factorizations, to run in polynomial-time. We refer to such proof-systems, in which the prover runs in polynomial-time given the NP witness, as *doubly-efficient proofs.*[1]

The question of doubly-efficient statistical zero-knowledge batch verification for prime products is interesting in its own right, but the same question is fascinating for a variety of cryptographic problems such as checking that a long list of public-keys/signatures/ciphertexts are all well-formed. This raises the natural question, posed already by Kaslasi *et al.* [KRR+20]:

Does every problem in $\mathrm{SZK} \cap \mathrm{NP}$ *have a* doubly-efficient statistical zero-knowledge batch verification proof*: namely, a statistical zero-knowledge proof that* $x_1, \ldots, x_k \in \Pi$ *with communication that scales poly-logarithmically in* k *and an honest-prover that runs in polynomial-time given the* k NP *witnesses?*

A source of hope, especially for our prime product example, are results by Reingold, Rothblum and Rothblum [RRR21, RRR18, RR20], which give doubly-efficient batch verification protocols for any problem in UP – the class of NP problems in which YES instances have unique witnesses. This shows that a rich class of problems in NP have batch verification protocols (indeed, the prime product problem is in UP, the witness is just the prime factorization). Unfortunately, the UP batch verification protocols in the foregoing works are *not* zero-knowledge (assuming $\mathrm{UP} \not\subseteq \mathrm{BPP}$) and, as a matter of fact, even explicitly reveal some of the witnesses to the verifier.

[1] Doubly-efficient proofs are also studied in the context of problems in P (rather than NP) in which case we require the prover to run in polynomial-time, without any additional auxiliary information. See the recent survey by Goldreich [Gol18].

1.1 Our Results

Our main result is a doubly-efficient statistical zero-knowledge batch proof for any problem in NISZK ∩ UP. Recall that by [MNRV24] such problems have a statistical zero-knowledge batch verification proof in which the honest prover is inefficient, and on the other hand, by [RR20], the same set of problems have a *different* doubly-efficient interactive proof, which is not zero-knowledge. Our main result shows how to combine these two protocols to obtain the best-of-both-worlds:

Theorem 1. *Let $k = k(n)$ such that $k(n) \leq 2^{n^{0.01}}$. Every problem $\Pi \in$ NISZK∩ UP has a public-coin* SZK *protocol for verifying k instances $x_1, \ldots, x_k$, each of length n, with the following parameters:*

- *Communication complexity:* $\mathrm{poly}(n, \log(k))$.
- *Number of rounds:* $\mathrm{polylog}(n, k)$.
- *Verifier runtime:* $\tilde{O}(k) \cdot \mathrm{poly}(n)$.
- *The honest prover, given also the k unique witnesses, runs in time* $\mathrm{poly}(n, k)$.

We emphasize that in contrast to general purpose results for succinct arguments (a la [Kil92]), the batching protocol of Theorem 1 does not rely on any unproven assumptions and yields the strong guarantees of *statistical* soundness and zero-knowledge.

Combining Theorem 1 with existing NISZK protocols from the literature, we immediately derive doubly-efficient statistical zero-knowledge batch verification protocols for several important problems. In particular, using the NISZK for prime products of [GMR98] we obtain a batch verification protocol for prime products (as well as variants such as quasi-safe prime products), which in particular yields an efficient method for batch verification of signature verification keys in the [GMR98] undeniable signature scheme. Using the fact that quadratic residuosity (of Blum Integers) as well as quadratic non-residuosity is in NISZK [BDSMP91], we similarly obtain such batch verification protocols for these problems.[2]

We remark that in contrast to the result of [MNRV24], our batching protocol is *interactive*. Obtaining a doubly-efficient NISZK batching protocol for NISZK∩ UP remains open. We elaborate on this in Sect. 1.3.

1.2 Technical Overview

Our starting point is the batch verification protocol for UP of Rothblum and Rothblum [RR20]. As noted above, this protocol meets all the requirements of

[2] For batch verification of QR, there is a simple reduction from k instances to 1 by taking a random modular subset product of the given integers. However, this only works in case we use the same modulus for each instance whereas our protocol works even when using different moduli. Also, we note that [BDSMP91] only give an NISZK for QNR. However, in case N is a Blum integer, QR reduces to QNR by multiplying the input by $-1 \pmod N$ (since -1 is a QNR modulo a Blum integer).

Theorem 1, except that it is blatantly *not* statistical zero-knowledge (assuming NISZK $\cap$ UP $\not\subseteq$ BPP) as it explicitly reveals some of the witnesses. Our goal is to transform the protocol to be statistical zero-knowledge, while preserving its complexity.

As the UP batching protocol of [RR20] is public-coin, a natural approach to convert it to be zero-knowledge is to utilize the "commit-and-prove" framework of Ben-Or *et al.* [BOGG+90]. This framework transforms public-coin protocols into zero-knowledge ones by letting the prover commit to her messages rather than sending them in the clear. Subsequently, the prover proves, using an additional zero-knowledge proof, that if she were to open the commitments, the original verifier would have accepted.

Trying to utilize this framework to our purposes we first run into the following problem: to attain a protocol that is simultaneously statistical zero-knowledge and statistically sound, we need commitments that are both statistically hiding and statistically binding. While such commitments do not exist per se, following [BMO90,IOS97,NV06,OV08] we can use an elegant relaxation of such commitments, called *instance-dependent commitments*.

An instance-dependent commitment scheme is a commitment scheme associated with an instance x of a promise problem Π. If x is a YES instance, the commitment is required to be hiding, and if x is a NO instance, then it should be binding. The above sequence of works utilized the fact that we only need binding to hold for NO instances, and hiding to hold for YES instances, and so such commitments suffice for implementing zero-knowledge proofs and in particular for implementing the commitments in the framework of [BOGG+90]. In particular, Nguyen and Vadhan [NV06] used (a suitable variant of) instance dependent commitments to show that any problem in SZK $\cap$ NP has a doubly-efficient SZK proof.

Ong and Vadhan [OV08] (building on [NV06]) constructed instance-dependent commitment schemes for any problem in SZK. Thus, we would like to start with the UP batching protocol of [RR20] and compile it to be zero-knowledge by utilizing the commit-and-prove approach, while using an instance dependent commitment. For a given input $x_1, \ldots, x_k$, since each of the individual parts x_i is an instance of our problem Π, which is in SZK (in fact NISZK), it has a corresponding instance dependent commitment. Using a standard combiner[3] for commitments we could then obtain a single instance dependent commitment for the batch verification problem. Alas, the length of commitments (and decommitments) obtained in this way scales linearly with k, which we cannot afford.

A better approach is to utilize the main result of Mu *et al.* [MNRV24], which gives a *statistical zero-knowledge* batch verification protocol for Π. Using their protocol, in combination with the instance dependent commitment characterization of [NV06,OV08] we obtain a *direct* instance-dependent commitment scheme for the batch problem $\Pi^{\otimes k} = \{(x_1, \ldots, x_k) : \forall i, x_i \in \Pi\}$. But what

[3] Here we need a combiner between k commitments, which needs to be hiding if all of them are hiding, and binding if at least one of them is binding. A suitable combiner in this setting is to simply commit separately using each commitment scheme.

are the lengths of commitments and decommitments in the resulting scheme? We observe that the main technical result of [MNRV24] can be interpreted as a reduction of k instances of an NISZK problem to a single instance of the *Image Density problem* [DSDCPY98]. In Image Density, the instances are circuits such that YES instances correspond to circuits that generate a distribution that is statistically close to uniform, whereas NO instances are circuits that generate a distribution with a relatively small support. While the size of the circuit generated by the [MNRV24] reduction scales (quasi-)linearly with k, the size of its *input and output* only grows poly-logarithmically in k. Our next observation is that the instance dependent commitment of [NV06] for Image Density has the feature that the lengths of the commitment and decommitment correspond only to *the input and output length of the circuit* rather than its size. Equipped with the above observation we obtain an instance-dependent commitment scheme for our batch verification problem with communication complexity $\mathrm{poly}(n, \log(k))$.

The above suffices for the "commit phase" of the transformation of Ben-Or *et al.* [BOGG+90] but for the "prove phase" we still need to run a zero-knowledge proof demonstrating that "had the prover opened the commitments, the verifier would have accepted". The naïve approach might be to use one of the classical zero-knowledge proofs for NP, such as the original proof due to Goldreich *et al.* [GMW91], while relying on the instance dependent commitment described above. Here however, we run into a major problem: the [GMW91] protocol, as well as all other zero-knowledge protocols in the literature, make non-blackbox use of the underlying circuit on which we prove correctness. In our case, this circuit incorporates the commitment's verification circuit, which includes the circuit C produced by the [MNRV24] reduction. As mentioned above, the circuit C has size $\mathrm{poly}(n, k)$. Therefore, this approach would result in an unacceptable communication complexity of $\mathrm{poly}(n, k)$.

To overcome this issue, we build on the recent work of Hazay *et al.* [HVW23], which used the "MPC-in-the-head" framework of Ishai *et al.* [IKOS09] in order to compile public-coin interactive protocols to be zero-knowledge *while using the commitment scheme as a blackbox*. We emphasize that while all zero-knowledge proofs in the literature need an explicit description of the *circuit*, there is no a priori need to make a non blackbox use of the underlying commitment. Indeed, Hazay *et al.* show how to compile several public-coin interactive proofs from the literature to be zero-knowledge in such a blackbox way.

We proceed to describe the transformation in more detail. Recall that the MPC-in-the-head paradigm offers a general transformation from Secure Multi-party Computation (MPC) protocols to zero-knowledge protocols. The high-level structure is as follows:

- Let $\Pi \in \mathrm{NP}$ be a promise problem with a corresponding witness relation $R_\Pi(x, w)$. We start with an MPC protocol for a functionality f that corresponds to verifying that the players' inputs form an additive secret sharing of a valid witness of R_Π (for some fixed x).

- The prover, "in-her-head" secret shares the witness amongst virtual parties and emulates the execution of the MPC protocol. Subsequently, she commits to the resulting views of the parties.
- The verifier selects a small subset of the views she wishes the prover to reveal.
- The prover decommits to these views.
- The verifier accepts only if the views are consistent (with one another and the protocol description) and are accepting.

We utilize the MPC-in-the-head paradigm to compile the [RR20] protocol to be zero-knowledge using the instance dependent commitment as a blackbox. The idea is to adjust our protocol so that the prover does not commit directly to her messages (as in [BOGG+90]). Rather, she commits to ℓ additive secret shares of each of her messages, for some small parameter ℓ. Then, the prover proceeds with an "MPC-in-the-head" proof, where she gives each simulated party a secret share of her messages. The computed functionality corresponds to the verification predicate used by the verifier in the UP batching protocol of [RR20].

This approach was inspired by the work of Hazay *et al.* [HVW23], and it only relies on black-box access to the circuit that verifies decommitments. Hence, since the circuit that verifies decommitments has input and output sizes $\mathrm{poly}(n, \log(k))$, our objective of achieving communication complexity at most $\mathrm{poly}(n, \log(k))$ remains feasible.

A final remaining problem is that the communication complexity of the MPC-in-the-head proof depends on the running time of the verifier for the UP batching protocol [RR20] (since the parties run an MPC protocol that emulates this computation). Unfortunately, this verifier has running time $k \cdot \mathrm{poly}(n, \log(k))$, which is prohibitively large (and is inherent since the verifier has to, at the very least, read its input). We solve this last issue by analyzing more closely the verifier of the [RR20] UP batching protocol. We observe that the verifier in their protocol consists of two main phases:

- A "preprocessing step" which depends only on the input and verifier's coin tosses. We emphasize that this step does not depend on the prover's messages and in particular can happen before the interaction begins. The verifier needs to only keep an internal state of size $\mathrm{poly}(n, \log(k))$ at the end of this step.
- An interactive step, in which the prover and verifier interact. The key point is that this step can be implemented in $\mathrm{poly}(n, \log(k))$ time.

To capitalize on this observation, we let both the verifier and the prover independently execute the preprocessing step. We then utilize the "MPC-in-the-head" proof only for the second step, which is computable in time $\mathrm{poly}(n, \log k)$, thereby obtaining communication complexity $\mathrm{poly}(n, \log k)$.

1.3 Open Questions

We highlight several natural follow-up questions that we leave open:

1. **Generalizing to** SZK**:** The key question left open by the line of work on batch verification of statistical zero-knowledge proofs is a general batch verification protocol that works for all of SZK, rather than just NISZK. Given such a result, it seems possible that our techniques, in combination with the UP batching protocol of [RR20], could yield a doubly-efficient SZK batch protocol for $\mathrm{SZK} \cap \mathrm{UP}$.
2. **Generalizing to** NP**:** Our result is limited to UP languages, where we inherit this limitation from [RR20]. The work of Bitansky *et al.* [BKP+23] indicates that a general doubly-efficient batch verification protocol for NP is somewhat unlikely, as it would lead to (unconditional) *statistical witness indistinguishable proofs* for NP. However, this barrier becomes meaningless (i.e., a non-issue) when considering languages in $\mathrm{SZK} \cap \mathrm{NP}$. Thus, there is no fundamental barrier that we are aware of in generalizing our results from $\mathrm{NISZK} \cap \mathrm{UP}$ to $\mathrm{NISZK} \cap \mathrm{NP}$ (or even $\mathrm{SZK} \cap \mathrm{NP}$ for that matter).
3. **Non-interactive:** Our protocol is designed to handle problems that have non-interactive proof-systems (namely, NISZK and UP). Unfortunately however, in contrast to the protocols that we start off with, our batch verification protocol is highly interactive. This is due to two reasons: first, the UP batching protocol of [RR20] which we use is highly interactive and second, the instance dependent commitment of [NV06] that we use is interactive. Thus, making our batch-verification protocol be non-interactive seems to require using fundamentally different techniques.

1.4 Organization

Preliminaries are in Sect. 2. The proof of the main result (namely, Theorem 1) is in Sect. 3. Some proofs are deferred to Appendices A to C.

2 Preliminaries

A *promise problem* Π consists of two disjoint sets of strings $\Pi = (\Pi_Y, \Pi_N)$. The set Π_Y is called the set of YES instances and the set Π_N is called the set of NO instances.

Definition 1. *The* statistical distance *between two random variables X, Y on a finite universe U, denoted by $\Delta(X,Y)$, is defined as:*

$$\Delta(X,Y) = \max_{S \subseteq U} \left(X\left(S\right) - Y\left(S\right)\right) = \frac{1}{2} \sum_{u \in U} \left|X(u) - Y(u)\right|.$$

2.1 Interactive Proofs and Zero-Knowledge

We use $\mathrm{view}_V\left(P\left(x\right), V(x)\right)$ to refer to the *view* of the verifier in an execution of an interactive protocol with prover P and verifier V on common input x. The view includes the input x, all messages sent by P to V in the protocol, and the verifier's random coin tosses. We say that the view is *accepting* if, at the end of the corresponding interaction, the verifier accepts.

Definition 2 (Interactive proof). *Let $c = c(n) \in [0,1]$ and $s = s(n) \in [0,1]$. An* interactive proof *with completeness error c and soundness error s for a promise problem Π, consists of a probabilistic polynomial-time verifier V and a computationally unbounded prover P such that following properties hold:*

- **Completeness:** *For any $x \in \Pi_Y$:*

$$\Pr\left[\text{view}_V\left(P\left(x\right), V(x)\right) \text{ is accepting}\right] \geq 1 - c(|x|).$$

- **Soundness:** *For any (computationally unbounded) cheating prover P^* and any $x \in \Pi_N$:*

$$\Pr\left[\text{view}_V\left(P^*\left(x\right), V(x)\right) \text{ is accepting}\right] \leq s(|x|).$$

We denote this proof system by (P, V).

An interactive proof (P, V) is *public-coin* if all the messages sent by the verifier are independent random strings (with a fixed length that is independent of the interaction). Let $\Pi = (\Pi_Y, \Pi_N) \in$ NP, with a corresponding witness relation $R_\Pi(x, w)$.

Definition 3 (Doubly-Efficient Interactive Proof). *Let Π be a promise problem. A* doubly-efficient interactive proof *for the relation R_Π is an interactive-proof (P, V) for Π in which the prover strategy P can be implemented in probabilistic polynomial-time given any* NP *witness $w \in \{w' : (x, w') \in R_\Pi\}$ as an auxiliary input.*

Zero-Knowledge. For the SZK definition, we allow the malicious verifier to have access to an auxiliary input $a \in \{0,1\}^*$. Accordingly, we also provide the simulator with the same auxiliary input a.

Definition 4 (SZK). *Let $z = z(n) \in [0,1]$. An interactive-proof (P, V) for Π is a* statistical zero-knowledge proof (SZK)*, with zero-knowledge error z, if for every probabilistic polynomial-time verifier V^* there exists a probabilistic polynomial-time algorithm Sim (called the* simulator*) such that for any $x \in \Pi_Y$ and $a \in \{0,1\}^*$:*

$$\Delta\left(\text{view}_{V^*}\left(P\left(x\right), V^*(x, a)\right), Sim(x, a)\right) \leq z(|x|).$$

If the completeness, soundness, and zero-knowledge errors are all negligible in $|x|$, we say that the interactive proof is an SZK proof. We also use SZK to denote the class of promise problems having SZK proofs.

Non-Interactive Statistical Zero-Knowledge. Next, we define the *non-interactive* variant of statistical zero-knowledge, denoted NISZK. As usual in this setting, we give the prover and verifier access to a uniformly generated common random string (CRS).

Definition 5 (NISZK). *Let $c = c(n) \in [0,1], s = s(n) \in [0,1]$ and $z = z(n) \in [0,1]$. A* non-interactive statistical zero-knowledge proof (NISZK) *with completeness error c, soundness error s and zero-knowledge error z for a promise problem Π, consists of a probabilistic polynomial-time verifier V, a computationally unbounded prover P and a polynomial $\ell = \ell(n)$ (the length of the* CRS*) such that the following properties hold:*

- **Completeness:** *For any $x \in \Pi_Y$:*

$$\Pr_{r \leftarrow \{0,1\}^{\ell(|x|)}} \left[V(x, r, P(x,r)) \text{ accepts} \right] \geq 1 - c(|x|).$$

- **Soundness:** *For any $x \in \Pi_N$:*

$$\Pr_{r \leftarrow \{0,1\}^{\ell(|x|)}} \left[\exists \pi^* \text{ s.t. } V(x, r, \pi^*) \text{ accepts} \right] \leq s(|x|).$$

- **Zero-Knowledge:** *There exists a probabilistic polynomial-time algorithm Sim (called the* simulator*) such that for any $x \in \Pi_Y$:*

$$\Delta\Big(\big(U_\ell, P\left(x, U_\ell\right)\big), Sim(x) \Big) \leq z(|x|),$$

  where U_ℓ denotes a random variable distributed uniformly over $\{0,1\}^{\ell(|x|)}$.

Unless stated otherwise, we assume that $c(\cdot), s(\cdot)$ and $z(\cdot)$ are negligible in $|x|$. We use NISZK to denote the class of promise problems having such an NISZK protocol.

2.2 Instance-Dependent Commitments for NISZK

Next, we recall the notion of instance-dependent commitments and revisit their construction for NISZK [NV06].[4] An *instance-dependent commitment* [BMO90, IOS97, NV06, OV08], for a promise problem Π, is a commitment scheme associated with an instance $x \in \{0,1\}^*$. Unlike standard commitment schemes, an instance-dependent commitment scheme requires the hiding property to hold only when x is a YES instance, and the binding property to hold only when x is a NO instance. We now provide a formal definition (the following definitions of instance-dependent commitments and their security are based on [OV08])

Definition 6 (Instance-dependent commitments). *An* instance-dependent commitment scheme *is a family of protocols $\{\mathrm{Com}_x\}_{x \in \{0,1\}^*}$ with the following properties:*

[4] The later work of Ong and Vadhan [OV08] constructs instance-dependent commitments for all of SZK. However, their construction is more complex and the simpler construction of instance-dependent commitments for NISZK, due to [NV06], suffices for our results.

- *Scheme* Com_x *proceeds in two stages: a* commit *stage and a* reveal *stage. In both stages, the* sender *and* receiver *receive instance* x *as common input, and hence we denote the sender and receiver as* S_x *and* R_x*, respectively, and write* $\mathrm{Com}_x = (S_x, R_x)$.
- *At the beginning of the commit stage, sender* S_x *receives a private input* $b \in \{0,1\}$*, which denotes the bit that* S_x *is supposed to commit to. At the end of the commit stage, both sender* S_x *and receiver* R_x *output a* commitment c.
- *In the reveal stage, sender* S_x *sends a pair* (b, d)*, where* d *is the* decommitment *string for bit* b*. Receiver* R_x *accepts or rejects based on* x*,* b*,* d*, and* c.
- *The sender* S_x *and receiver* R_x *algorithms are computable in polynomial time (in* $|x|$*), given* x *as auxiliary input.*
- *For every* $x \in \{0,1\}^*$*,* R_x *will always accept (with probability 1) if both sender* S_x *and receiver* R_x *follow their prescribed strategy.*

The instance-dependent commitment scheme $\{\mathrm{Com}_x = (S_x, R_x)\}_{x\in\{0,1\}^*}$ *is* public coin *if for every* $x \in \{0,1\}^*$*, all messages sent by* R_x *are independent random coins.*

To simplify notation, we write Com_x or (S_x, R_x) to denote the instance-dependent commitment scheme $\{\mathrm{Com}_x = (S_x, R_x)\}_{x\in\{0,1\}^*}$. The hiding and binding properties of standard commitments extend in a natural way to their instance-dependent analogs.

Definition 7 (Hiding of instance-dependent Commitments). *Let* $\varepsilon = \varepsilon(n) \in [0,1]$*. The instance-dependent commitment scheme* $\mathrm{Com}_x = (S_x, R_x)$ *is* ε-statistically hiding on $I \subseteq \{0,1\}^*$ *if for every* R^**, and every* $x \in I$*,*

$$\Delta\left(\mathrm{view}_{R^*}\left(S_x(0), R^*\right), \mathrm{view}_{R^*}\left(S_x(1), R^*\right)\right) \leq \varepsilon(|x|),$$

where random variable $\mathrm{view}_{R^*}(S_x(b), R^*)$ *denotes the view of* R^* *in the commit stage after interacting with* $S_x(b)$*. For a problem* $\Pi = (\Pi_Y, \Pi_N)$*, an instance-dependent commitment scheme* Com_x *for* Π *is* ε-statistically hiding on the YES instances *if* Com_x *is* ε*-statistically hiding on* Π_Y*.*

Definition 8 (Binding of instance-dependent Commitments). *Let* $\varepsilon = \varepsilon(n) \in [0,1]$*. The instance-dependent commitment scheme* $\mathrm{Com}_x = (S_x, R_x)$ *is* ε-statistically binding on $I \subseteq \{0,1\}^*$ *if for every* S^**, and for all* $x \in I$*, the malicious sender* S^* *succeeds in the following game with probability at most* $\varepsilon(|x|)$*:*

1. *The sender* S^* *interacts with* R_x *in the commit stage obtaining commitment* c.
2. *Then* S^* *outputs pairs* $(0, d_0)$ *and* $(1, d_1)$*, and* succeeds *if in the reveal stage,* $R_x(0, d_0, c) = R_x(1, d_1, c) = \mathrm{accept}$.

For a problem $\Pi = (\Pi_Y, \Pi_N)$*, an instance-dependent commitment scheme* Com_x *for* Π *is* ε-statistically binding on the NO instances *if* Com_x *is* ε*-statistically binding on* Π_N*.*

Remark 1. We defined instance-dependent commitment schemes as bit commitments. However, they can be extended to string commitments with the same round complexity by executing multiple bit commitments in parallel.

We revisit the instance-dependent commitment scheme for all of NISZK, given in [NV06]. This scheme is for the NISZK-complete promise problem Image Density[5] [DSDCPY98, GSV99]. Since Image Density is NISZK-complete, this commitment scheme can be adapted to suit any NISZK promise problem. We first define the Image Density problem.

Definition 9. *Let* $\varepsilon = \varepsilon(n) \in [0,1]$. *The* ε-Image Density *problem* $(\mathrm{ID}_\varepsilon)$, *is defined as follows:*

$$\mathrm{ID}_{\varepsilon,Y} = \{C : \Delta(U_\mu, C) \leq \varepsilon\},$$
$$\mathrm{ID}_{\varepsilon,N} = \{C : |\mathrm{Supp}(C)| \leq \varepsilon \cdot 2^\mu\},$$

where C *is a circuit of size* n *with* η *input bits and* μ *output bits. We highlight that we use the unconventional symbols* η *and* μ, *as* n *and* m *will be used later for alternative purposes.*

That is, YES instances of Image Density have an output distribution that is statistically close to uniform, and NO instances of Image Density have an output distribution with a small support. The next lemma, due to [NV06], shows that Image Density has an instance-dependent commitment.

Lemma 1 ([NV06]). *Let* $\varepsilon = \varepsilon(n) \in [0,1]$ *and let* n, η, μ *denote the size, the input length, and the output length of an* ID_ε *circuit, respectively. The problem* ID_ε *has an instance-dependent commitment scheme that is* 2ε*-statistically hiding on the* YES *instances and* $(\mathrm{poly}(\mu) \cdot \varepsilon)$*-statistically binding on the* NO *instances. Furthermore, the number of bits sent between the sender and the receiver during the commit and reveal phases is* $\mathrm{poly}(\eta, \mu)$, *the receiver runs in time* $n \cdot \mathrm{poly}(\eta, \mu, \log n)$, *and the commitment scheme is public coin and constant-round.*

We emphasize that the poly in the furthermore clause above is a fixed polynomial and in particular does not depend on the *size* of the circuit. While the furthermore clause of Lemma 1 follows from the construction of Nguyen and Vadhan [NV06], it is not explicitly stated therein. Hence, to give precise bounds on the communication complexity, we provide a proof of Lemma 1 in Appendix B.

[5] The commitment scheme, as presented in [NV06], was for a different promise problem known as EA. However, it involved a preprocessing step intended to transform an EA instance into an Image Density instance. Since we will work directly with Image Density instances, we skip this preprocessing step.

2.3 Batch Protocols

We introduce the concept of batch problems and review known batching results for UP and for NISZK [RR20,MNRV24].

Given a promise problem Π, we define the promise problem $\Pi^{\otimes k}$ that consists of k equal-sized instances of Π. We denote by n the size of each instance.

Definition 10. *For a given promise problem Π and an integer k, the problem $\Pi^{\otimes k} = (\Pi_Y^{\otimes k}, \Pi_N^{\otimes k})$ is defined as follows:*

$$\Pi_Y^{\otimes k} = \{(x_1, \ldots, x_k) : \forall i \in [k], x_i \in \Pi_Y, |x_1| = \ldots = |x_k|\}, \text{ and}$$
$$\Pi_N^{\otimes k} = \left\{(x_1, \ldots, x_k) \in (\Pi_Y \cup \Pi_N)^k : \exists j \in [k], x_j \in \Pi_N, |x_1| = \ldots = |x_k|\right\}.$$

2.3.1 Batching for NISZK

We introduce a lemma from [MNRV24]. We note that this lemma is not directly mentioned in [MNRV24], but can be deduced by examining the proofs of [MNRV24, Theorem 1.1] and [MNRV24, Lemma 3.9]. We analyze these proofs to establish this lemma in Appendix A.

Lemma 2 ([MNRV24]). *Let $\Pi \in$ NISZK and $k = k(n)$ such that $k(n) \leq 2^{n^{0.01}}$. Then, $\Pi^{\otimes k}$ has a randomized Karp reduction to ID_ε with the following properties:*

- *The reduction is computable in time $k \cdot \mathrm{poly}(n, \log k)$ and uses $\mathrm{poly}(n, \log k)$ random coins.*
- *The reduction never errs on NO instances and errs only with probability negligible in n and k on YES instances.*
- *ε is negligible in both n and k.*
- *The ID_ε circuit produced by the reduction has size $k \cdot \mathrm{poly}(n, \log k)$, with input and output sizes $\mathrm{poly}(n, \log k)$.*

2.3.2 Doubly Efficient Batching for UP

Next we present the batch verification result for UP, due to [RR20].

Theorem 2 ([RR20, Corollary 5]). *For every promise problem $\Pi \in$ UP, there exists a public-coin interactive proof, with perfect completeness and soundness error $\frac{1}{2}$, for verifying that k instances $x_1, \ldots, x_k$, each of length n, are all in Π. The complexity of the protocol is as follows:*

- *Communication complexity:* $\mathrm{poly}(n, \log(k))$.
- *Number of rounds:* $\mathrm{polylog}(n, k)$.
- *Verifier runtime:* $\tilde{O}(n \cdot k) + \mathrm{poly}(n, \log(k))$.
- *The honest prover, given the k unique witnesses, runs in time* $\mathrm{poly}(n, k)$.

Furthermore, before the interaction begins, the verifier runs a pre-processing step on inputs $x_1, \ldots, x_k$ and the public coins to be sent to the prover. The pre-processing time is $\tilde{O}(n \cdot k)$, and its output is a string pp of length $\mathrm{poly}(n, \log(k))$. *Following the interaction, the verifier runs in time* $\mathrm{poly}(n, \log(k))$ *on inputs pp and the prover messages and decides whether to accept or reject.*

Remark 2. The results in [RR20] are stated for languages, but readily extend to the setting of promise problems. More importantly, the "furthermore" part of Theorem 2 is not explicitly stated in [RR20], but can be inferred by inspecting their protocol, see Appendix C for details.

2.4 Secure Multiparty Computation

Our protocol relies on the MPC-in-the-head framework of Ishai *et al.* [IKOS09]. In this section we provide the relevant background following [IKOS09]. Since it suffices for our purposes, we consider *semi-honest* MPC protocols, with perfect security, for functionalities of a particular form.

Let ℓ be the number of players, which will be denoted by $P_1, ..., P_\ell$. We will be interested in computations in which all players share a public input x, and each player P_i holds a local private input w_i. We consider protocols that securely realize an ℓ-party functionality f, where f maps the joint input $(x, w_1, \ldots, w_\ell)$ to a single output bit b.

A protocol Π is specified via its next message function. That is, the function $\Pi\left(i, x, w_i, r_i, (m_1, \ldots, m_j)\right)$ returns the set of ℓ messages sent by P_i in round $j+1$ given the public input x, its local input w_i, its random input r_i, and the messages $m_1, \ldots, m_j$ it *received* in the first j rounds. The output of Π may also indicate that the protocol should terminate, in which case Π returns the local output of P_i. The *view* of P_i, denoted by view_i, includes w_i, r_i, and the messages *received* by P_i during the execution of Π. Note that the messages sent by an uncorrupted player P_i as well as its local output can be inferred from view_i and x by invoking the next message function of Π. It will be useful to define the following natural notion of consistency between views.

Definition 11 (Consistent views). *We say that two views* view_i *and* view_j *are* consistent *(for the protocol* Π *and the public input* x*) if the outgoing messages implicit in* (view_i, x) *are identical to the incoming messages reported in* view_j *and vice versa.*

The following lemma asserts that an ℓ-tuple of views corresponds to some honest execution of Π if and only if every pair of views is consistent.

Lemma 3 ([IKOS09, Lemma 2.3]). *Let* Π *be an* ℓ*-party protocol as above and* x *be a public input. Let* $\text{view}_1, \ldots, \text{view}_\ell$ *be an* ℓ*-tuple of (possibly incorrect) views. Then all pairs of views* $(\text{view}_i, \text{view}_j)$ *are consistent for* Π *and* x *if and only if there exists an honest execution of* Π *with public input* x *(and some choice of private inputs* w_i *and random inputs* r_i*) in which* view_i *is the view of* P_i *for every* $1 \leq i \leq \ell$*.*

In the semi-honest model, one may break the security requirements into the following correctness and privacy requirements.

Definition 12 (Correctness). *We say that the protocol* Π *realizes a deterministic* ℓ*-party functionality* $f\left(x, w_1, \ldots, w_\ell\right)$ *with* perfect correctness *if for all*

inputs $x, w_1, \ldots, w_\ell$*, the probability that the output of some player is different from the output of* f *is* 0*, where the probability is over the independent choices of the random inputs* $r_1, \ldots, r_\ell$*.*

Definition 13 (*t*-Privacy). *Let* $1 \leq t < \ell$*. We say that* Π *realizes* f *with* perfect t-privacy *if there is a probabilistic polynomial-time simulator Sim such that for any inputs* $x, w_1, \ldots, w_\ell$ *and every set of corrupted players* $T \subseteq [\ell]$*, where* $|T| \leq t$*, the joint view* $\text{view}_T(x, w_1, \ldots, w_\ell)$ *of players in* T *is distributed identically to* $Sim\left(T, x, (w_i)_{i \in T}, f_T(x, w_1, \ldots, w_\ell)\right)$*.*

We now define a notion of efficiency for protocols that securely realize functionalities computable by circuits.

Definition 14 (Efficiency). *Let* f *be an* ℓ*-party functionality computed by a circuit* C *of size* s*, and let* Π *realize* f*. We say that* Π *is* C-efficient *if every player* P_i *runs in time* $\text{poly}(|x|, |w_i|, \ell, s)$*, during the entire execution of* Π*.*

In our batch proof, we utilize an MPC protocol as part of the MPC-in-the-head paradigm. As an example, we will instantiate our batch proof with the renowned BGW MPC protocol [BGW88] to demonstrate that our proof is realizable.

Theorem 3 ([BGW88, Theorem 1], see also [AL17]). *For every function* f *computable by a circuit* C*, there exists a* C*-efficient MPC protocol* Π *with* 5 *players, that realizes* f *with perfect correctness and perfect* 2*-privacy.*

3 Doubly Efficient Zero-Knowledge Batching for NISZK ∩ UP

In this section, we prove Theorem 1 by showing a doubly-efficient statistical zero-knowledge batch proof for problems in NISZK∩UP. We begin by combining the results of [MNRV24] and [NV06], to show that batch instances of NISZK have an instance-dependent commitment scheme with communication complexity that scales poly-logarithmically with k.

Lemma 4. *Let* $\Pi \in$ NISZK *and* $k = k(n)$ *such that* $k(n) \leq 2^{n^{0.01}}$*. Then,* $\Pi^{\otimes k}$ *has an instance-dependent commitment scheme that is* ε*-statistically hiding on the* YES *instances and* ε*-statistically binding on the* NO *instances, with* ε *being negligible in* n *and in* k*. The number of bits sent between the sender and the receiver during the commit and reveal phases is* $\text{poly}(n, \log k)$*. Furthermore, the commitment scheme is public-coin and constant-round, and the receiver runs in time* $k \cdot \text{poly}(n, \log k)$*.*

We recall that an instance $(x_1, \ldots, x_k) \in \Pi^{\otimes k}$ is considered a YES instance if each x_i is a YES instance of Π, and is considered a NO instance if at least one the x_i is a NO instance of Π.

Proof. The lemma follows by composing the commitment scheme for Image Density of Lemma 1 with the randomized Karp reduction from $\Pi^{\otimes k}$ to Image Density of Lemma 2.

Thus, on common input $(x_1, \ldots, x_k)$, the sender and the receiver reduce $(x_1, \ldots, x_k)$ to an ID_ε circuit C. This only requires the sender to send the randomness that was used for the reduction, which is of size $\mathrm{poly}(n, \log k)$, to the receiver. Then, the sender and the receiver use C as the common input for the commitment scheme of Lemma 1 and proceed with the commit and reveal phases as per usual. The commitment scheme is ε-statistically hiding on the YES instances since the reduction of Lemma 2 fails only with negligible probability when the sender is honest. The commitment scheme is ε-statistically binding on the NO instances since the reduction of Lemma 2 never fails on NO instances.

Remark 3. Our main protocol involves commitments to multiple bits. We could in principle improve the efficiency of the protocol by reducing $(x_1, \ldots, x_k)$ to an ID_ε instance just once rather than with each commitment, but we avoid this optimization for the sake of simplicity.

We now have everything we need to construct the doubly-efficient SZK batch proof for NISZK $\cap$ UP, thereby establishing Theorem 1.

3.1 Proof of Theorem 1

We first present the protocol and then prove why it meets all the requirements of Theorem 1.

Consider the UP batching protocol $(P_{\mathrm{UP}}, V_{\mathrm{UP}})$, promised by Theorem 2. In this protocol, the verifier V_{UP} runs in two phases. In the preprocessing phase, the verifier runs on inputs $x_1, \ldots, x_k$ and the public coins to be sent to the prover, and produces a string pp of length $\mathrm{poly}(n, \log k)$ (we emphasize that in this phase there is no interaction). In the online phase, the verifier interacts with the prover and decides whether to accept based only on pp and the prover's messages. In this phase, the verifier runs in time $\mathrm{poly}(n, \log k)$. Denote the verifier messages by $\overline{\alpha} = (\alpha_1, \ldots, \alpha_r)$, where α_i is the verifier's message in round i, and similarly denote the prover's messages by $\overline{\beta} = (\beta_1, \ldots, \beta_r)$. Let $C_{V_{\mathrm{UP}}}$ be a size $\mathrm{poly}(n, \log k)$ circuit that computes the verifier's decision predicate. That is, $C_{V_{\mathrm{UP}}}$ operates on inputs pp and the prover's messages and determines whether to accept.

Let f be the following function, that takes as input a string pp, and ℓ additive shares of $\overline{\beta}$ (the value of ℓ will be determined later):

$$f\left(pp, \overline{\beta^1}, \ldots, \overline{\beta^\ell}\right) = C_{V_{\mathrm{UP}}}\left(pp, \overline{\beta^1} \oplus \ldots \oplus \overline{\beta^\ell}\right),$$

where $\oplus$ denotes the bitwise exclusive-or operation:

$$\overline{\beta^i} \oplus \overline{\beta^j} = \left(\beta_1^i \oplus \beta_1^j, \ldots, \beta_r^i \oplus \beta_r^j\right).$$

Let Π be an ℓ-player $C_{V_{\mathrm{UP}}}$-efficient[6] MPC protocol, with a constant $\ell = 5$, that realizes f with perfect correctness and perfect (semi-honest) 2-privacy, i.e., the protocol of Theorem 3. The public input is pp, and $\overline{\beta^i}$ is the private input of player P_i. Upon completion of Π, the local output of all players should match the output of f.

Using the MPC protocol from above, together with the protocol of Theorem 2, We now describe the zero-knowledge batch verification protocol in Fig. 1, where the commitment scheme that is used throughout the protocol is the one of Lemma 4.

We analyze the protocol to verify that it fulfills all the requirements of Theorem 1.

Completeness. Suppose $x_1, \ldots, x_k \in \Pi_Y$. The perfect completeness of the underlying UP batching protocol and the perfect correctness of Π ensure that the shares $\overline{\beta^i}$ that P commits to during Step 1 would lead player P_i, running on public input pp and private input $\overline{\beta^i}$, to output 1 with probability 1. The views that P commits to are consistent with their respective players' inputs and with each other, so V would accept if P successfully revealed all the required commitments. Since the commitment scheme of Lemma 4 has correctness as long as the sender and the receiver follow their prescribed strategies, perfect completeness of our protocol follows.

Soundness. Suppose at least one of $x_1, \ldots, x_k$ is in Π_N and fix a (computationally unbounded) cheating prover P^*. Without loss of generality, let us assume that P^* is deterministic. By Lemma 4, the commitment scheme used throughout the protocol is statistically binding, with binding error negligible in both n and k. We delay addressing the binding error for now, by first analyzing the soundness for a modified protocol (P', V'), which we will now describe.

In this modified protocol, commitments are not used and the prover P' sends all her messages openly. Accordingly, the verifier V' does not verify decommitments on Step 6a, but only ensures that the messages sent in Step 5 are consistent with the previous messages sent by P'.

The prover P' simulates the adversary P^* and acts as an intermediary between P^* and V'. Since P^* is the adversary for the unmodified protocol, she is expected to use commitments, whereas V' does not. Thus, the prover P' will manipulate the communication as follows. The prover P' will act as the receiver of the commitments and decommitments made by P^*. After each commitment is done, P' uses its unbounded computational power to extract the message from the commitment. If the commitment is invalid or if it can be opened ambiguously, a default message is selected instead. After extracting the message, P' sends it in the clear to V'. Whenever P' receives a message from V', she feeds it

[6] We slightly abuse notation here since $C_{V_{\mathrm{UP}}}$ does not compute the functionality f (Note that f has $\ell + 1$ inputs while $C_{V_{\mathrm{UP}}}$ only has two). This abuse of notation is justified since the sizes of $C_{V_{\mathrm{UP}}}$ and the closely related circuit that computes f differ only by a multiplicative factor of $\mathrm{poly}(\ell)$, and ℓ is constant.

Common Inputs: $x_1, \ldots, x_k \in \{0,1\}^n$.
Prover's Additional Inputs: $w_1, \ldots, w_k$, which are the unique witnesses for $x_1, \ldots, x_k$.

The protocol employs the commitment scheme described in Lemma 4 and the MPC protocol of Theorem 3.

The Protocol:

1. The prover P and the verifier V execute the protocol of Theorem 2, with the following modification: Each time the prover P is supposed to send a message β_i in the clear, she instead additively secret shares her message into ℓ shares $\beta_i = \beta_i^1 \oplus \ldots \oplus \beta_i^\ell$. She then commits to each of those shares separately. Crucially, since the protocol of Theorem 2 is public-coin, the verifier V can disregard the modification in the prover's messages and continue to send random coins as usual.
2. The prover P first collects the shares from the previous step:

$$\forall i \in [\ell] : \overline{\beta^i} \triangleq \left(\beta_1^i, \ldots, \beta_r^i\right)$$

   Subsequently, the prover and the verifier, each on their own, run the preprocessing step of V_{UP} promised by Theorem 2 on inputs $x_1, \ldots, x_k$ and the verifier's messages to obtain pp. Then, the prover simulates the MPC protocol Π (defined earlier in the text) with public input pp and player P_i having private input $\overline{\beta^i}$. This yields the ℓ views of the ℓ players.
3. The prover commits separately to the view of each of the ℓ players.
4. The verifier selects two players $i \neq j$ at random and asks the prover to reveal their views.
5. The prover decommits to $\left(\left(\beta_1^i, \ldots, \beta_r^i\right), \mathrm{view}_i\right)$ and $\left(\left(\beta_1^j, \ldots, \beta_r^j\right), \mathrm{view}_j\right)$.
6. The verifier accepts if and only if all the following hold true:
   (a) The prover has successfully revealed all the required commitments.
   (b) The public input in view_i is pp, the private input is $\left(\beta_1^i, \ldots, \beta_r^i\right)$.
   (c) The public input in view_j is pp, the private input is $\left(\beta_1^j, \ldots, \beta_r^j\right)$.
   (d) The views $(\mathrm{view}_i, \mathrm{view}_j)$ are consistent, and the players output 1 in both.

Fig. 1. Doubly-efficient SZK batching protocol for NISZK $\cap$ UP

to P^*. This describes the modifications in (P', V'). We move on to analyze the soundness of the modified protocol.

Recall $C_{V_{\mathrm{UP}}}$, a circuit that operates on inputs pp and the prover's messages and computes the V_{UP} verifier's decision predicate. Given the soundness of the underlying UP batching protocol of Theorem 2, the messages $\overline{\beta}$ that P' sends during Step 1, would make the verifier V_{UP} reject with probability at least $\frac{1}{2}$. That is, $\Pr\left[C_{V_{\mathrm{UP}}}\left(pp, \overline{\beta}\right) = 0\right] \geq \frac{1}{2}$. Assume we are in the case where $C_{V_{\mathrm{UP}}}\left(pp, \overline{\beta}\right) = 0$. Therefore, since the MPC protocol Π has perfect correctness,

an honest execution of Π with public input pp and private inputs $\overline{\beta^i}$ results in all players outputting 0, in which case V' rejects on Step 6d with probability 1.

An honest execution of the MPC protocol Π with different inputs will lead V' to reject in Steps 6b or 6c with probability at least $\frac{1}{\ell}$. If the execution is not honest, by Lemma 3, there exists a pair of inconsistent views, and V' selects it with probability at least $\frac{1}{\binom{\ell}{2}}$, causing V' to reject in Step 6d.

Considering $\Pr\left[C_{V_{\mathrm{UP}}}\left(pp, \overline{\beta}\right) = 0\right] \geq \frac{1}{2}$, we get soundness error $1 - \frac{1}{2\binom{\ell}{2}}$ for (P', V'). We now factor in the binding error. If for some verifier coins of V' together with some receiver coins of P', V' rejects after interacting with P' and all the commitments that P' simulates are binding, then the same coins, when used by the verifier V, cause V to reject after interacting with P^*. Thus, a union bound yields a statistical soundness error of $1 - \frac{1}{2\binom{\ell}{2}} + \delta(n, k)$, where δ is negligible in both n and k.

Taking ℓ to be a constant, we have obtained a constant soundness error for the protocol. Later, we will reduce the error to be negligible by (sequential) repetition (see Remark 4).

Zero Knowledge. Let V^* be a probabilistic polynomial-time verifier. Since the MPC protocol Π is perfectly (semi-honest) 2-private, it has a simulator Sim_{MPC}. Using V^* and Sim_{MPC}, we construct a simulator Sim_{ZK} for our protocol in Fig. 2. The commitment scheme that is used throughout the simulation is the instance-dependent commitment of Lemma 4. We note that the simulator Sim_{ZK} only utilizes V^* in a black-box manner, and recall that black-box zero-knowledge implies auxiliary input zero-knowledge [GO94].

The zero-knowledge analysis closely follows that of [GMW91,IKOS09], and we outline it here for the sake of completeness. Suppose $x_1, \ldots, x_k \in \Pi_Y$. By Lemma 4, the commitment scheme used throughout the simulation will be statistically hiding, with hiding error negligible in both n and k. To demonstrate that the output distribution of the simulator is statistically close to that of real verifier views, we examine several hybrid distributions, wherein the simulator is given the witnesses to $x_1, \ldots, x_k$:

1. Distributions $A_0, \ldots, A_t$: In distribution A_i, during the first i attempts, the simulator Sim_{ZK} acts like the honest prover in Step 1, committing to shares of the prover's messages instead of committing to random shares. In the rest of the steps, the simulator proceeds as usual. In the remaining attempts, the simulator follows its standard strategy during all steps.
   Since any number of secret shares smaller than ℓ distributes uniformly and independently, and because the commitments hide all shares except the two shares that are revealed, the statistical distance between each $A_i, A_i + 1$ is negligible in n and k.
2. Distributions $B_0, \ldots, B_t$: In all of these distributions, the simulator behaves like the honest prover in Step 1, committing to shares of the prover's messages instead of committing to random shares. In distribution B_i, during the first i attempts, instead of utilizing the MPC simulator Sim_{MPC}, the

The Simulator $Sim_{\mathrm{ZK}}(x_1, \ldots, x_k)$**:**

Attempt $t = \ell^2 \cdot \left(\log^2(n) + \log^2(k)\right)$ times:

1. Simulate Step 1 of the protocol with V^*, with the following modification: Whenever the prover is supposed to commit to shares corresponding to message β_i, commit to random shares instead.
2. Compute pp according to $x_1, \ldots, x_k$ and the verifier messages $\overline{\alpha}$ obtained from the previous step.
3. Randomly select a pair of players $i \neq j$, and run the MPC simulator accordingly. Namely, compute $Sim_{\mathrm{MPC}}\left(\{i,j\}, pp, \left(\overline{\beta^i}, \overline{\beta^j}\right), 1\right)$ to get the views of the players i, j.
4. Generate arbitrary views for the other players (of the right length) and feed V^* with commitments to all of the generated views.
5. The verifier V^* responds with a request that the views of the players $\{i', j'\}$ be revealed. If $\{i,j\} = \{i',j'\}$, the attempt succeeded. Consequently, The simulator reveals the requested views and outputs the view of V^*. If $\{i,j\} \neq \{i',j'\}$, the attempt failed, and we start over.

If all attempts fail, output $\perp$.

Fig. 2. Simulator for the protocol of Theorem 1

simulator Sim_{ZK} executes the MPC protocol Π to obtain the views of the two randomly selected players. Then, the simulator chooses the views of the remaining players arbitrarily and proceeds as usual. In the other attempts, the simulator follows its standard strategy during all steps except for Step 1 (in which it mimics the honest prover).
Note that $B_0 = A_t$. The statistical distance between each $B_i, B_i + 1$ is exactly zero due to the perfect 2-privacy of the MPC protocol Π.

3. Distribution C: In this distribution, the simulator behaves like in distribution B_t, but instead of generating random views for the players that are not in $\{i,j\}$, it assigns them the views computed when executing the MPC protocol Π. The statistical distance between B_t, C is negligible in n and k since the commitments to the views of the players not in $\{i,j\}$ are never revealed, and the commitment scheme is statistically hiding.

The only difference between distribution C and that of real verifier views is that C can output $\perp$ when all attempts fail. In the sampling of the distribution C, each attempt succeeds with a probability of at least $\frac{1}{\binom{\ell}{2}}$. Given that attempts are independent, the simulator succeeds on at least one attempt with all but negligible probability in n and k. Therefore, the statistical distance between C and that of real views is negligible in n and k.

Since A_0 is the output distribution of the simulator Sim_{ZK}, we deduce that the statistical distance between the output distribution of Sim_{ZK} and that of real verifier views is negligible in n and k, as required.

Complexity. We begin with analyzing the communication complexity of the protocol. The commitment scheme of Lemma 4 increases the communication complexity by a multiplicative factor of only $\text{poly}(n, \log k)$ per committed bit, both for the commit and the reveal phases. Hence, for the sake of analysis, we can conveniently overlook this overhead.

- **Step 1:** The verifier's messages and prover's messages during Step 1 of the protocol have length $\text{poly}(n, \log k)$ by Theorem 2.
- **The remaining steps:** The circuit $C_{V_{\text{UP}}}$ has size $\text{poly}(n, \log k)$, and the MPC protocol Π is $C_{V_{\text{UP}}}$-efficient. Therefore, the overall size of the MPC players' views is $\ell \cdot \text{poly}(n, \log(k), \ell) = \text{poly}(n, \log(k), \ell)$, and this term dominates the communication complexity of the remaining steps.

Since our parameter ℓ is a constant, the total communication complexity is $\text{poly}(n, \log k)$.

We proceed to analyze the round complexity of the protocol. The commitment scheme of Lemma 4 is constant-round. Therefore, in Step 1, The protocol inherits its round complexity $\text{polylog}(n, k)$ from the underlying UP batching protocol of Theorem 2. During the remaining steps, the protocol proceeds with an additional constant number of rounds, resulting in a total round complexity of $\text{polylog}(n, k)$.

The verifier's runtime is $k \cdot \text{poly}(n, \log(k))$ due to the efficiency of the verifier of Theorem 2, the efficiency of the receiver of Lemma 4, and the $C_{V_{\text{UP}}}$-efficiency of the MPC protocol Π.

The prover runs in time $\text{poly}(n, k)$ given the k unique witnesses, since the protocol of Lemma 2 is doubly-efficient, and because the MPC protocol Π is $C_{V_{\text{UP}}}$-efficient.

Lastly, the protocol inherits its public-coin nature from the underlying UP batching protocol of Theorem 2 and the commitment scheme of Lemma 4.

Remark 4. In our analysis, we only achieved a constant soundness error. Our zero-knowledge simulator only makes black-box use of the verifier, so by sequential composition (see [GO94]), we can repeat our proof $\text{poly}(\log(n), \log(k))$ times to get a negligible soundness error while preserving zero-knowledge and maintaining the complexity of the proof as previously stated.

Acknowledgements. Or Keret and Ron Rothblum are funded by the European Union (ERC, FASTPROOF, 101041208). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them. Prashant Nalini Vasudevan is supported by the National Research Foundation, Singapore, under its NRF Fellowship programme, award no. NRF-NRFF14-2022-0010.

A Proof of Lemma 2

Before we prove Lemma 2, we revisit the relevant definitions and results that appear in [MNRV24]. We start with some basic definitions of probability theory.

Definition 15. *Let X be a random variable distributed over a universe U, and for every $x \in U$, denote by $p_x = \Pr[X = x]$. We recall the following notions of entropy of X:*

- $H_0(X) = \log(|\{x : p_x \neq 0\}|)$.
- $H_1(X) = -\sum_{x \in U} p_x \log(p_x)$.
- $H_2(X) = -\log(\mathrm{cp}(X))$.[7]

We also define the following smoothed notion of entropy version utilized in [MNRV24].

Definition 16 (Smooth Entropy [RW04]). *For any $\varepsilon \geq 0$, the* ε-smooth H_2 entropy *of a random variable X is defined as follows:*

$$H_2^{\varepsilon}(X) = \max_{Y \in \mathcal{B}_{\varepsilon}(X)} H_2(Y),$$

where $\mathcal{B}_{\varepsilon}(X)$ is the set of all distributions within statistical distance ε of X.

We recall the definition of the *Smooth Entropy Approximation* problem considered in [MNRV24].

Definition 17. *Let $\varepsilon = \varepsilon(n) \in [0, 1]$. The* ε-Smooth Entropy Approximation *problem* ($\mathrm{SEA}_{\varepsilon}$)*, is defined as follows:*

$$\mathrm{SEA}_{\varepsilon,Y} = \{(C, k) : H_2^{\varepsilon}(C) \geq k + 1\},$$

$$\mathrm{SEA}_{\varepsilon,N} = \{(C, k) : H_0(C) \leq k - 1\},$$

where C is a circuit with η input bits, $\mu \leq 3\eta$ output bits, and $0 < k \leq \mu$.

That is, YES instances of $\mathrm{SEA}_{\varepsilon}$ are close to a distribution that has high H_2 entropy, and NO instances of $\mathrm{SEA}_{\varepsilon}$ have low H_0 entropy (i.e., a small support). We now introduce a lemma based on the work [MNRV24]. While this lemma is not stated explicitly in [MNRV24], it follows immediately from the proof of [MNRV24, Theorem 1.1].

Lemma 5 ([MNRV24, See Proof of Theorem 1.1]). *Let $\Pi \in$ NISZK and $k = k(n)$ such that $k(n) \leq 2^{n^{0.01}}$. Then, $\Pi^{\otimes k}$ has a randomized Karp reduction to $\mathrm{SEA}_{\varepsilon}$ with the following properties:*

- *The reduction is computable in time $k \cdot \mathrm{poly}(n, \log k)$ and uses $\mathrm{poly}(n, \log k)$ random coins.*
- *The reduction never errs on NO instances and errs only with probability negligible in n and k on YES instances.*
- *ε is negligible in both n and k.*

[7] Recall that the collision probability of a distribution X is defined as $\mathrm{cp}(X) = \Pr_{x,x' \leftarrow X}[x = x']$.

– *The* SEA_ε *circuit* C *generated by the reduction has size* $k \cdot \mathrm{poly}(n, \log k)$*, with input and output sizes* $\mathrm{poly}(n, \log k)$*.*

Remark 5. The fact that the running time of the reduction and the size of the SEA_ε circuit C are both $k \cdot \mathrm{poly}(n, \log k)$ does not follow from the proof of [MNRV24, Theorem 1.1], but rather it can be inferred from the reduction that establishes [MNRV24, Theorem 5.1].

We now have all the necessary details from [MNRV24] to establish Lemma 2. The reduction from $\Pi^{\otimes k}$ to $\mathrm{ID}_{\varepsilon'}$ operates as follows (note that we use ε' for the ID circuits and ε for the SEA circuits since these two parameters will be different). It first efficiently reduces the tuple $(x_1, \ldots, x_k)$ to an SEA_ε instance using the reduction described in Lemma 5. Subsequently, it reduces the SEA_ε instance to an $\mathrm{ID}_{\varepsilon'}$ circuit $\widehat{C}$.

We now present the reduction from SEA_ε to $\mathrm{ID}_{\varepsilon'}$. This is a deterministic Karp reduction, and its analysis closely follows [MNRV24, Lemma 3.9]. Denote by (C, κ) the SEA_ε instance. The circuit C has η input bits and μ output bits. Let $H_{\mu,\kappa} = \{h : \{0,1\}^\mu \to \{0,1\}^\kappa\}$ be a pairwise-independent family of hash functions as in [MNRV24, Lemma 2.9]. Each hash function from this family is described by $O(\max(\mu, \kappa)) = O(\mu)$ bits. Construct the circuit C' that corresponds to $r = 20(\log^2 n + \log^2 k)$ copies of C evaluated independently. Its input length is $\eta' = \eta \cdot r$, and its output length is $\mu' = \mu \cdot r$. Similarly, let $\kappa' = \kappa \cdot r$. The reduction, on input (C, κ), outputs a circuit $\widehat{C}$ that works as follows:

– It takes as input a description h of a hash function in $H_{\mu',\kappa'}$ and an $x \in \{0,1\}^{\eta'}$.
– It outputs $\Big(h, h\big(C'(x)\big)\Big)$.

The output length of $\widehat{C}$ is $\hat{\mu} = O(\mu') + \kappa' < O(\max(\eta', \mu'))$. Its input length is also $O(\max(\eta', \mu'))$. Suppose (C, κ) is a YES instance of SEA_ε. That is, $H_2^\varepsilon(C) \geq \kappa + 1$, and thus $H_2^{\varepsilon'}(C') \geq \kappa' + r$, where $\varepsilon' = \varepsilon \cdot r$. This implies that there is a distribution Y that is at most ε'-far from C' that has $\mathrm{cp}(Y) \leq 2^{-(\kappa'+r)}$. Let H denote the random variable corresponding to a uniformly random $h \in H_{\mu',\kappa'}$. By the leftover hash lemma (see [MNRV24, Lemma 2.10] for the exact formulation), the statistical distance between $(H, H(Y))$ and $(H, U_{\kappa'})$ is at most $2^{(-r)/2}$. Thus, the distance between $(H, H(C'))$ and $(H, U_{\kappa'})$ is at most $\varepsilon' + 2^{(-r)/2} = \varepsilon \cdot 20\left(\log^2 n + \log^2 k\right) + 2^{-10\left(\log^2 n + \log^2 k\right)}$. Since ε is negligible in n and k, $\Delta\left(\widehat{C}, U_{\hat{\mu}}\right)$ is also negligible in both n and k.

On the other hand, suppose (C, κ) is a NO instance of SEA_ε, that is, $H_0(C) \leq \kappa - 1$. This means that C has support of size at most $2^{\kappa-1}$, and C' has support of size at most $2^{\kappa'-r}$. This implies that the support size of $(H, H(C'))$ is at most $|H_{\mu',\kappa'}| \cdot 2^{\kappa'-r} \leq 2^{-r} \cdot 2^{\hat{\mu}}$. Therefore, $\left|\mathrm{Supp}\left(\widehat{C}\right)\right| \cdot 2^{-\hat{\mu}} = 2^{-r}$ is negligible in both n and k.

Since the hash functions have a succinct description and are efficiently computable, the running time of the reduction and the size of the circuit $\widehat{C}$ are

both $k \cdot \text{poly}(n, \log k)$, and $\widehat{C}$ has input and output lengths $O(\max(\eta', \mu')) = \text{poly}(n, \log k)$, as required. This completes the analysis of the reduction from SEA_ε to $\text{ID}_{\varepsilon'}$. By combining the reduction of Lemma 5 with the reduction from SEA_ε to $\text{ID}_{\varepsilon'}$, we have proved Lemma 2.

B Proof of Lemma 1

We prove Lemma 1 by first presenting the instance-dependent commitment scheme of [NV06], and then demonstrating that it satisfies the conditions of Lemma 1.

The commitment scheme makes use of *interactive hashing* [NOVY98, DHRS07], specifically employing an information-theoretically secure protocol from [DHRS07]. We begin by defining interactive hashing.

Interactive Hashing. In an interactive hashing protocol, two players are participating, A and B. Player A receives an input W, while B has no input. Upon executing the protocol, both A and B output a pair (W_0, W_1), such that one of W_0, W_1 equals W. Informally, the protocol is secure for A if, when W is uniformly distributed, even a computationally unbounded B cannot determine which one of (W_0, W_1) equals W. The protocol is secure for B if, for any sufficiently sparse set S, even a computationally unbounded A cannot force both W_0 and W_1 to reside in S. We now provide the formal definitions, based on [NV06].

Definition 18 (Interactive Hashing). *A protocol (A, B) is called an* interactive hashing protocol *if it is an* efficient *two-party protocol with the following properties:*

- **Inputs:** *A has an input string $W \in \{0,1\}^\mu$ and B has no input.*
- **Outputs:** *A and B output two distinct values $W_0, W_1 \in \{0,1\}^\mu$ (in lexicographic order) such that one of W_0, W_1 equals W.*

Definition 19. *Let D denote the distribution of the index $d \in \{0,1\}$ such that the string W_d corresponds to the input of A in the interactive hashing protocol. An interactive hashing protocol is* secure *for A if for every unbounded B^* the distributions $\{\text{view}_{B^*}(A(W), B^*), D\}$ and $\{\text{view}_{B^*}(A(W), B^*), U_1\}$ are identical when $W \equiv U_\mu$.*

*An interactive hashing protocol is (δ, ρ)-*secure *for B if for every $S \subseteq \{0,1\}^\mu$ of density at most δ and every computationally unbounded strategy A^*, it holds that $\Pr[W_0, W_1 \in S] < \rho$.*

We will rely on an interactive hashing protocol due to Ding *et al.* [DHRS07].

Lemma 6 ([NV06, Theorem 4.3], based on [DHRS07]). *For every $0 < \delta < 1$, there exists a constant-round public-coin interactive hashing protocol (A, B) that is secure for A and $(\delta, \text{poly}(\mu) \cdot \delta)$-secure for B.*

Using Lemma 6, we now present in Fig. 3 the construction of the instance-dependent commitment for ID_ε. We proceed to analyze the construction to show that it satisfies Lemma 1. (Although our focus is on the communication complexity, for completeness we provide a full analysis).

Common Input: Circuit $C : \{0,1\}^\eta \to \{0,1\}^\mu$.
Sender's Additional Input: a bit $b \in \{0,1\}$.

Commit phase:

1. The sender S generates a random $r \in \{0,1\}^\eta$ and computes $x = C(r)$
2. (S, R) run the interactive hashing protocol (A, B) of Lemma 6 with the parameter $\delta = \varepsilon$, and with S playing $A(x)$ and R playing B. Their common output is a pair (x_0, x_1).
3. The sender S finds $d \in \{0,1\}$ such that $x_d = x$ and sends $c = d \oplus b$.
4. The commitment z is defined as (x_0, x_1, c).

Reveal phase:

1. S reveals b and r.
2. The receiver R accepts if $C(r) = x_{c \oplus b}$ and otherwise it rejects.

Fig. 3. Instance-dependent commitment scheme for ID_ε

Complexity. During the commit phase, the sender samples a random output x from the circuit C and engages in an interactive hashing protocol with the receiver on input x. Since both of the parties in the interactive hashing protocol are efficient and C has output size μ, the number of exchanged bits is $\mathrm{poly}(\mu)$. Subsequently, the sender sends one additional bit to the receiver. During the reveal phase, the sender sends the revealed bit along with the input r to C that was used to generate x. The circuit C has input size η, thus $\mathrm{poly}(\eta, \mu)$ bits are exchanged in both the commit and the reveal phases.

During the commit phase, the receiver runs in time $\mathrm{poly}(\eta, \mu)$, due to the interactive hashing protocol of [DHRS07] being efficient. During the reveal phase, the receiver runs in time $|C| \cdot \mathrm{poly}(\eta, \mu, \log(|C|))$ to evaluate the circuit C on input r. Hence, the receiver has runtime $|C| \cdot \mathrm{poly}(\eta, \mu, \log(|C|))$ in both phases.

Also note that the commitment scheme is constant-round and public-coin since the interactive hashing protocol of [DHRS07] is constant-round and public-coin.

Correctness. Given any input circuit and bit b, if the sender and the receiver follow their prescribed strategies, then the receiver always accepts. This is ensured by the underlying interactive hashing protocol of Lemma 6 which guarantees that one of the output strings (x_0, x_1) will be equal to the input x.

Hiding Error. In case C is a YES instance of ID_ε, then $\Delta(C, U_\mu) \leq \varepsilon$. For every receiver R^*, we denote by B^* the interactive hashing strategy induced by R^*. Additionally, we denote by $d \in \{0,1\}$ the index of the interactive hashing output that equals the input (to A). For a random variable X and an index $b \in \{0,1\}$,

we write as a shorthand $v(X,b) \triangleq (\text{view}_{B^*}(A(X), B^*), b)$. We then have:

$$\begin{aligned}\Delta(\text{view}_{R^*}(S(0),R^*), \text{view}_{R^*}(S(1),R^*)) &= \Delta(v(C,d\oplus 0), v(C,d\oplus 1))\\ &\le \Delta(v(C,d), v(U_\mu,d))\\ &\quad + \Delta(v(U_\mu,d), v(U_\mu,d\oplus 1))\\ &\quad + \Delta(v(U_\mu,d\oplus 1), v(C,d\oplus 1)))\\ &\le \Delta(C,U_\mu) + 0 + \Delta(C,U_\mu)\\ &\le 2\varepsilon.\end{aligned}$$

Therefore, the hiding error $\Delta\big(\text{view}_{R^*}(S(0),R^*), \text{view}_{R^*}(S(1),R^*)\big)$ is at most 2ε.

Binding Error. In case C is a NO instance of ID_ε, the density of $\text{Supp}(C)$ is at most ε. Let S^* be a malicious sender participating in the game defining the binding property. The sender S^* can succeed only if both x_0, x_1 are in the support of C. Because we chose the parameter δ for the interactive hashing protocol to match the density of $\text{Supp}(C)$, by the $(\delta, \text{poly}(\mu)\cdot\delta)$-security of the interactive hashing we have $\Pr[x_0, x_1 \in \text{Supp}(C)] < \text{poly}(\mu)\cdot\delta$. Since $\delta = \varepsilon$, the probability of violating the binding condition is at most $\text{poly}(\mu)\cdot\varepsilon$.

C More Details on Theorem 2

The [RR20] batching proof (similarly to many interactive proofs in the literature, such as the sumcheck and [GKR15] protocols) is "holographic". This means that the verifier runs in time $\text{poly}(n, \log(k))$ given oracle access to the *low degree extension* (LDE) of the input $(x_1, \ldots, x_k)$ (see [GR17] for a formal treatment). Moreover, the locations of the oracle queries (to the LDE) of the input only depend on the verifier's internal randomness. Thus, to see that the furthermore part of Theorem 2 holds, observe that the verifier can compute these values during its pre-processing step (i.e., prior to its interaction with the prover).

To see that the [RR20] protocol is indeed holographic we briefly recall the construction. Their protocol (building on an idea from [RRR18]) is recursive, where in each step we reduce the task of batch proving k instances, to batch proving $k/2$ related instances. The reduction relies on an *interactive proof of proximity* (IPP) which is developed in the same work (improving on a prior result of [RVW13]). The reduction step proceeds by running an IPP whose input is a list of the k witnesses concatenated with an LDE of the k instances. The IPP verifier queries some points from this LDE (which is why we view the protocol as holographic) and needs to additionally query some points of the witnesses. The parameters are set so that the verifier only needs to query at most $k/2$ of the witnesses, so rather than actually performing these queries, we recursively check these via an additional batch verification protocol.[8]

[8] To facilitate the recursion, we need to rely on a protocol that does batch verification and additionally checks some (small-depth) predicate on the k witnesses.

References

[AL17] Asharov, G., Lindell, Y.: A full proof of the BGW protocol for perfectly secure multiparty computation. J. Cryptol. **30**(1), 58–151 (2017)

[BDSMP91] Blum, M., De Santis, A., Micali, S., Persiano, G.: Noninteractive zero-knowledge. SIAM J. Comput. **20**(6), 1084–1118 (1991)

[BGW88] Ben-Or, M., Goldwasser, S., Wigderson, A.: Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In: Simon, J. (eds.) Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2–4, 1988, Chicago, Illinois, USA, pp. 1–10. ACM (1988)

[BKP+23] Bitansky, N., Kamath, C., Paneth, O., Rothblum, R., Vasudevan, P.N.: Batch proofs are statistically hiding. Electron. Colloquium Comput. Complex., TR23-077 (2023)

[BMO90] Bellare, M., Micali, S., Ostrovsky, R.M.: Perfect zero-knowledge in constant rounds. In: Symposium on the Theory of Computing (1990)

[BOGG+90] Ben-Or, M., et al.: Everything provable is provable in zero-knowledge. In: Goldwasser, S. (ed.) CRYPTO 1988. LNCS, vol. 403, pp. 37–56. Springer, New York (1990). https://doi.org/10.1007/0-387-34799-2_4

[DHRS07] Ding, Y.Z., Harnik, D., Rosen, A., Shaltiel, R.: Constant-round oblivious transfer in the bounded storage model. J. Cryptol. **20**, 165–202 (2007)

[DSDCPY98] De Santis, A., Di Crescenzo, G., Persiano, G., Yung, M.: Image density is complete for non-interactive-SZK. In: Larsen, K.G., Skyum, S., Winskel, G. (eds.) ICALP 1998. LNCS, vol. 1443, pp. 784–795. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0055102

[GKR15] Goldwasser, S., Kalai, Y.T., Rothblum, G.N.: Delegating computation: interactive proofs for muggles. J. ACM **62**(4), 27:1–27:64 (2015)

[GMR98] Gennaro, R., Micciancio, T., Rabin, D.: An efficient non-interactive statistical zero-knowledge proof system for quasi-safe prime products. In: 5th ACM Conference on Computer and Communication Security (CCS'98), pp. 67–72, San Francisco, California, November 1998. ACM, ACM Press (1998)

[GMW91] Goldreich, O., Micali, S., Wigderson, A.: Proofs that yield nothing but their validity for all languages in NP have zero-knowledge proof systems. J. ACM **38**(3), 691–729 (1991)

[GO94] Goldreich, O., Oren, Y.: Definitions and properties of zero-knowledge proof systems. J. Cryptol. **7**(1), 1–32 (1994). https://doi.org/10.1007/BF00195207

[Gol18] Goldreich, O.: On doubly-efficient interactive proof systems. Found. Trends Theor. Comput. Sci. **13**(3), 158–246 (2018)

[GR17] Gur, T., Rothblum, R.D.: A hierarchy theorem for interactive proofs of proximity. In: Papadimitriou, C.H. (ed.) 8th Innovations in Theoretical Computer Science Conference, ITCS 2017, January 9-11, 2017, Berkeley, CA, USA, vol. 67, LIPIcs, pp. 39:1–39:43. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2017)

[GSV99] Goldreich, O., Sahai, A., Vadhan, S.: Can statistical zero knowledge be made non-interactive? or on the relationship of SZK and *NISZK*. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 467–484. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48405-1_30

[HVW23] Hazay, C., Venkitasubramaniam, M., Weiss, M.: Beyond MPC-in-the-head: black-box constructions of short zero-knowledge proofs. In: Rothblum, G., Wee, H. (eds.) TCC 2023. LNCS, vol. 14369, pp. 3–33. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-48615-9_1

[IKOS09] Ishai, Y., Kushilevitz, E., Ostrovsky, R., Sahai, A.: Zero-knowledge proofs from secure multiparty computation. SIAM J. Comput. **39**(3), 1121–1152 (2009)

[IOS97] Itoh, T., Ohta, Y., Shizuya, H.: A language-dependent cryptographic primitive. J. Cryptol. **10**(1), 37–49 (1997). https://doi.org/10.1007/s001459900018

[Kil92] Kilian, J.: A note on efficient zero-knowledge proofs and arguments (extended abstract). In: Kosaraju, S.R., Fellows, M., Wigderson, A., Ellis, A.J. (eds.) Proceedings of the 24th Annual ACM Symposium on Theory of Computing, May 4-6, 1992, Victoria, British Columbia, Canada, pp. 723–732. ACM (1992)

[KRR+20] Kaslasi, I., Rothblum, G.N., Rothblum, R.D., Sealfon, A., Vasudevan, P.N.: Batch verification for statistical zero knowledge proofs. In: Pass, R., Pietrzak, K. (eds.) TCC 2020. LNCS, vol. 12551, pp. 139–167. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-64378-2_6

[KRV21] Kaslasi, I., Rothblum, R.D., Vasudevanr, P.N.: Public-coin statistical zero-knowledge batch verification against malicious verifiers. In: Canteaut, A., Standaert, F.-X. (eds.) EUROCRYPT 2021. LNCS, vol. 12698, pp. 219–246. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-77883-5_8

[MNRV24] Mu, C., Nassar, S., Rothblum, R., Vasudevan, P.N.: Strong batching for non-interactive statistical zero-knowledge. Electron. Colloquium Comput. Complex. TR24–024 (2024)

[NOVY98] Naor, M., Ostrovsky, R., Venkatesan, R., Yung, M.: Perfect zero-knowledge arguments for NP using any one-way permutation. J. Cryptol. **11**, 87–108 (1998)

[NV06] Nguyen, M.-H., Vadhan, S.: Zero knowledge with efficient provers. In: Proceedings of the Thirty-Eighth Annual ACM Symposium on Theory of Computing, pp. 287–295 (2006)

[OV08] Ong, S.J., Vadhan, S.: An equivalence between zero knowledge and commitments. In: Canetti, R. (ed.) TCC 2008. LNCS, vol. 4948, pp. 482–500. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78524-8_27

[RR20] Rothblum, G.N., Rothblum, R.D.: Batch verification and proofs of proximity with polylog overhead. In: Pass, R., Pietrzak, K. (eds.) TCC 2020. LNCS, vol. 12551, pp. 108–138. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-64378-2_5

[RRR18] Reingold, O., Rothblum, G.N., Rothblum, R.D.: Efficient batch verification for UP. In: Servedio, R.A. (eds.) 33rd Computational Complexity Conference, CCC 2018, June 22–24, 2018, San Diego, CA, USA, volume 102 of LIPIcs, pp. 22:1–22:23. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2018)

[RRR21] Reingold, O., Rothblum, G.N., Rothblum, R.D.: Constant-round interactive proofs for delegating computation. SIAM J. Comput. **50**(3) (2021)

[RVW13] Rothblum, G.N., Vadhan, S.P., Wigderson, A.: Interactive proofs of proximity: delegating computation in sublinear time. In: Boneh, D., Roughgarden, T., Feigenbaum, J., (eds.) Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013, pp. 793–802. ACM (2013)

[RW04] Renner, R., Wolf, S.: Smooth Rényi entropy and applications. In: International Symposium on Information Theory, 2004. ISIT 2004. Proceedings, p. 233. IEEE (2004)

Monotone Policy BARGs from BARGs and Additively Homomorphic Encryption

Shafik Nassar[1(✉)], Brent Waters[1,2], and David J. Wu[1]

[1] UT Austin, Austin, TX, USA
shafik@cs.utexas.edu
[2] NTT Research, Sunnyvale, CA, USA

Abstract. A monotone policy batch NP language $\mathcal{L}_{\mathcal{R},P}$ is parameterized by a monotone policy $P\colon \{0,1\}^k \to \{0,1\}$ and an NP relation $\mathcal{R}$. A statement $(x_1,\ldots,x_k)$ is a YES instance if there exists $w_1,\ldots,w_k$ where $P(\mathcal{R}(x_1,w_1),\ldots,\mathcal{R}(x_k,w_k)) = 1$. For example, we might say that an instance $(x_1,\ldots,x_k)$ is a YES instance if a majority of the statements are true. A monotone policy batch argument (BARG) for NP allows a prover to prove that $(x_1,\ldots,x_k) \in \mathcal{L}_{\mathcal{R},P}$ with a proof of size $\mathsf{poly}(\lambda, |\mathcal{R}|, \log k)$, where λ is the security parameter, $|\mathcal{R}|$ is the size of the Boolean circuit that computes $\mathcal{R}$, and k is the number of instances. Recently, Brakerski, Brodsky, Kalai, Lombardi, and Paneth (CRYPTO 2023) gave the first monotone policy BARG for NP from the learning with errors (LWE) assumption.

In this work, we describe a generic approach for constructing monotone policy BARGs from any BARG for NP together with an additively homomorphic encryption scheme. This yields the first constructions of monotone policy BARGs from the k-Lin assumption in prime-order pairing groups as well as the (subexponential) DDH assumption in *pairing-free* groups. Central to our construction is a notion of a zero-fixing hash function, which is a relaxed version of a predicate-extractable hash function from the work of Brakerski et al. Our relaxation enables a direct realization of zero-fixing hash functions from BARGs for NP and additively homomorphic encryption, whereas the previous notion relied on leveled homomorphic encryption, and by extension, the LWE assumption.

As an application, we also show how to combine a monotone policy BARG with a puncturable signature scheme to obtain a monotone policy aggregate signature scheme. Our work yields the first (statically-secure) monotone policy aggregate signatures that supports general monotone Boolean circuits from standard pairing-based assumptions. Previously, this was only known from LWE.

1 Introduction

A non-interactive batch argument (BARG) for NP allows a prover to convince a verifier that a collection of k statements $x_1,\ldots,x_k$ is true with a proof whose size scales *sublinearly* with k. Beyond the immediate application to amortizing

E. Boyle and M. Mahmoody (Eds.): TCC 2024, LNCS 15365, pp. 399–430, 2025.
https://doi.org/10.1007/978-3-031-78017-2_14

the communication cost of NP verification, batch arguments for NP also play a key role in constructing delegation for RAM programs (also known as a succinct non-interactive argument (SNARG) for P) [KVZ21, CJJ21b, KLVW23] and incrementally verifiable computation [DGKV22, PP22]. These objects have received extensive study recently, and to date, we have constructions from most standard algebraic assumptions in cryptography such as the learning with errors (LWE) assumption [CJJ21b], the k-Lin assumption on groups with bilinear maps [WW22], the (sub-exponential) decisional Diffie-Hellman (DDH) assumption in pairing-free groups [CGJ+23], or combinations of quadratic residuosity and (sub-exponential) DDH in pairing-free groups [CJJ21a, HJKS22].

Beyond batch NP *and* P. The recent successes in constructing succinct arguments for batch NP and for P from standard cryptographic assumptions has motivated the study of other (sub)-classes of NP for which we can build succinct non-interactive arguments from standard (falsifiable) assumptions. Very recently, Brakerski, Brodsky, Kalai, Lombardi, and Paneth [BBK+23] showed how to construct SNARGs for monotone policy batch NP. At a high level, the monotone policy batch NP language $\mathcal{L}_{\mathcal{R},P}$ is defined with respect to an NP relation $\mathcal{R}$ together with a monotone policy $P\colon \{0,1\}^k \to \{0,1\}$ as follows:

$$\mathcal{L}_{\mathcal{R},P} = \{(x_1,\ldots,x_k) \mid \exists (w_1,\ldots,w_k) : P(\mathcal{R}(x_1,w_1),\ldots,\mathcal{R}(x_k,w_k)) = 1\}\,.$$

In words, an instance $(x_1,\ldots,x_k)$ is true as long as an acceptable subset of the statements are true (as determined by the policy P). Such "monotone policy batch arguments" capture policies like majority, general thresholds, and more. The standard batch argument corresponds to the special case where the policy P is a simple conjunction.

Brakerski et al. [BBK+23] provided two constructions of monotone policy BARGs for NP. The first construction only relies on standard (somewhere extractable) BARGs and collision-resistant hash functions, but could only support monotone policies of logarithmic depth (i.e., monotone NC^1). To extend to monotone policies of arbitrary polynomial depth, they combine standard BARGs with a new notion of a predicate-extractable hash function, which they then build from the LWE assumption (specifically, they rely on leveled homomorphic encryption). This yields a monotone policy batch argument for arbitrary monotone policies from the LWE assumption. Due to the current reliance on leveled homomorphic encryption to construct the predicate-extractable hash function, instantiations of monotone policy BARGs for arbitrary-depth policies rely on the LWE assumption.

1.1 Our Results

Our main result in this work is showing how to construct BARGs for monotone policies by combining a (standard) BARG with an *additively* homomorphic encryption scheme (which can in turn be built from most number-theoretic assumptions [Gam84, Pai99, Reg05]). Combined with the recent progress on constructing BARGs from pairing-based groups [WW22] and pairing-free

groups [CGJ+23], we obtain the first monotone policy BARGs for NP from the k-Lin assumption over pairing groups and from the (sub-exponentially) DDH assumption in pairing-free groups. We provide an overview of our techniques in Sect. 1.2 and summarize our main results in the following theorem:

Theorem 1 (Informal). *Assuming any of (1) the plain LWE assumption, (2) the k-Lin assumption over pairing groups for any constant $k \in \mathbb{N}$, or (3) the (sub-exponential) DDH assumption in pairing-free groups, there exists a monotone policy BARG for all polynomial-size monotone circuit policies. The monotone policy BARG satisfies non-adaptive soundness and the proof size is $\mathsf{poly}(\lambda + |C| + \log |P|)$, where $|C|$ denotes the size of the Boolean circuit computing the NP relation, and $|P|$ is the size of the monotone policy.*

Monotone Policy Aggregate Signatures. A key difference between Theorem 1 and the previous LWE-based construction [BBK+23] is that we obtain a *non-adaptively-sound* BARG for monotone circuit policies whereas the [BBK+23] construction satisfied a stronger "somewhere extractability" notion. That is, in [BBK+23], the common reference string (CRS) can be sampled in a trapdoor mode and the trapdoor can be used to recover a witness for some x_i given a valid proof on statements $(x_1, \ldots, x_k)$. While extractability is often useful to have in a cryptographic primitive, it is not always essential.

As an illustrative example, we show how to use monotone policy BARGs in conjunction with (puncturable) signatures [GVW19] to construct a monotone policy aggregate multi-signature scheme. In an aggregate multi-signature scheme, there is a set of k signers, each with a signing/verification key-pair $(\mathsf{sk}_i, \mathsf{vk}_i)$. Given a policy P and a set of signatures σ_i for $i \in S$ (where σ_i verifies with respect to vk_i) on a common message m, if the set S satisfies the policy P, then it is possible to aggregate $\{\sigma_i\}_{i \in S}$ into a single short signature whose size is sublinear in $|S|$. For instance, P might encode a "threshold" policy that accepts all sets of size at least t. Crucially, static security of our monotone policy aggregate signature scheme only relies on *non-adaptive soundness* of the monotone policy BARG and security of the puncturable signature scheme. There is no need for an explicit extraction requirement. Very briefly, a puncturable signature scheme allows one to sample a "punctured" verification key vk (and associated signing key) for some message m^*. The punctured verification key is computationally indistinguishable from a normal verification key, but has the property that there does not *exist* any signatures on the punctured message m^* with respect to the punctured key. As shown in [GVW19], puncturable (or "all-but-one signatures") can be constructed from many standard number-theoretic assumptions. We summarize this result in the following theorem:

Theorem 2 (Informal). *Assuming the existence of a non-adaptively sound monotone BARG and a puncturable signature scheme, there exists a monotone policy aggregate multi-signature scheme. The scheme satisfies static unforgeability and the size of the aggregate signature is $\mathsf{poly}(\lambda + \log |P|)$, where $|P|$ denotes the size of the circuit computing the monotone policy.*

Theorem 2 shows that in combination with puncturable signatures, soundness alone is sufficient for building aggregate signatures for general monotone policies. Notably, Theorem 2 also provides the *first* monotone policy aggregate signature from pairing-based assumptions (in the plain model). Previous work have shown how to build *vanilla* aggregate signatures using (vanilla) non-interactive batch arguments [WW22,DGKV22]. In an independent and concurrent work, [BCJP24] also show how to construct a monotone policy aggregate multi-signature. Their work provides two constructions of monotone policy aggregate (multi)-signatures. The first scheme supports monotone policies that can be implemented by a read-once, bounded-space Turing machine and is also *adaptively* secure. This scheme relies on somewhere extractable BARGs and a verifiable private information retrieval scheme [BKP22], and can be instantiated from standard pairing-based or lattice-based assumptions. The second scheme supports policies implemented by an arbitrary monotone Boolean circuit, but achieves a weaker security definition (closer to static security) and also relies on fully homomorphic encryption (which to date, is not known from pairing-based assumptions). Theorem 2 gives a statically-secure monotone policy aggregate signature scheme that supports all monotone Boolean circuits, and does not rely on fully homomorphic encryption. This enables a new instantiation from pairings.

Soundness vs. Extraction. While Theorem 2 shows that extraction is *unnecessary* for all applications of monotone policy BARGs, our proof strategy for arguing soundness can nonetheless be extended to achieve a notion of extractability (see the full version [NWW23, Section 8]). The notion we achieve is similar to the somewhere extractability notion from [BBK+23], where for every monotone policy P, they define a notion of a "necessary set" associated with P (i.e., a set with the property that for every satisfying input $(x_1, \ldots, x_n)$ to P, there exists $i \in S$ where $x_i = 1$). The somewhere extractability notion from [BBK+23] programs S into the common reference string, and asserts that whenever the prover comes up with an accepting proof for statements $(x_1, \ldots, x_k)$ for an NP relation $\mathcal{R}$ and policy P, then the extractor will output w_i for $i \in S$ where $\mathcal{R}(x_i, w_i) = 1$. Our construction satisfies a looser variant of this property where the success probability of the extractor is smaller by a factor of $1/k$. We refer to this notion as semi-somewhere extractability. While our construction does achieve this notion of extraction with essentially no modification (see the full version [NWW23, Section 8]), we choose to focus on the simpler notion of non-adaptive soundness in the core part of this paper. Our rationale is twofold:

- First, there is a lack of consensus on what the "right" notion of extraction is when it comes to the setting of monotone policy BARGs. Notably, the recent and concurrent work of [BCJP24] that builds monotone policy aggregate signatures highlighted the *inadequacy* of the somewhere extractability notion from [BBK+23] for their particular application to constructing monotone policy aggregate signatures. Indeed, the work of [BCJP24] propose two different and seemingly incomparable notions of extraction for their applica-

tion. This illustrates that the most useful or desirable notion of extraction for monotone policy BARGs may be application-dependent.
– Second, while it is straightforward to show that our construction satisfies some notion of extractability, proving this property does not appear to confer additional capabilities. For the main application to statically-secure aggregate signatures, we showed above that non-adaptive soundness already suffices. There is no need for extraction if this is the end goal. The main advantage of having some kind of extractability definition is we can apply this construction to compile any digital signature scheme into a monotone policy aggregate signature scheme, as opposed to restricting ourselves to puncturable signatures (and we show this in the full version of this paper [NWW23]). While there is a qualitative benefit to this, we do not view it as strong evidence that semi-somewhere extractability is a clearly more powerful or more useful notion than non-adaptive soundness.

A New Application: general-policy BARGs for $\mathsf{NP} \cap \mathsf{coNP}$**.** We also highlight a simple application of BARGs for monotone policy batch NP to constructing a BARG that supports *arbitrary* policies over languages in $\mathsf{NP} \cap \mathsf{coNP}$. Our observation essentially follows the similar strategy of extending monotone closure of SZK to non-monotone closure [Vad06]. Specifically, for a language $\mathcal{X} \in \mathsf{NP} \cap \mathsf{coNP}$ and an arbitrary policy $P\colon \{0,1\}^k \to \{0,1\}$, we define the language

$$\mathcal{L}_{\mathcal{X},P} = \{(x_1, \ldots, x_k) \mid P(b_1, \ldots, b_k) = 1 \text{ where } b_i = \mathbb{1}\{x_i \in \mathcal{X}\}\},$$

where $\mathbb{1}\{x_i \in \mathcal{X}\}$ is the indicator function that outputs 1 if $x_i \in \mathcal{X}$ and 0 otherwise. Importantly, in this context, we allow P to be any arbitrary (possibly non-monotone) Boolean circuit. It is not difficult to see that a BARG for monotone policy batch NP immediately implies a BARG for $\mathcal{L}_{\mathcal{X},P}$. Namely, we first re-express the circuit P on k inputs $b_1, \ldots, b_k$ as a new *monotone* circuit P' on $2k$ inputs corresponding to the original input bits $b_1, \ldots, b_k$ as well as their negations $\bar{b}_1, \ldots, \bar{b}_k$. We can then apply a BARG for monotone policy batch NP on the set of $2k$ inputs with the policy P'. For this transformation to work, it is important that for each statement x_i, the prover can either provide a proof of membership $x_i \in \mathcal{X}$ (which sets $b_i = 1$) or a proof of non-membership $x_i \notin \mathcal{X}$ (which sets $\bar{b}_i = 1$).

1.2 Technical Overview

The starting point of our BARG construction is the "canonical protocol" from [BBK+23, §2.1]. We recall this below. In our description, we will consider the NP relation of Boolean circuit satisfiability.

– Given a Boolean circuit $C\colon \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$, a monotone policy $P\colon \{0,1\}^k \to \{0,1\}$, statements $x_1, \ldots, x_k \in \{0,1\}^n$, witnesses $w_1, \ldots, w_k \in \{0,1\}^h$, the prover first computes $b_i \leftarrow C(x_i, w_i)$ for all $i \in [k]$.

- The prover then evaluates the circuit $P(b_1, \ldots, b_k)$. The prover commits to all of the wire values in $P(b_1, \ldots, b_k)$ using a succinct commitment com that supports local openings. We index the input wires with the integers $1, \ldots, k$, the output wire by s (where s is the number of wires in P), and the intermediate wires with $k+1, \ldots, s-1$.
- The prover uses a batch argument to prove the following statements with respect to the commitment com:
  - **Input wires:** For every input wire $j \in [k]$, it proves that there exists a local opening of com to a value $b_j \in \{0,1\}$ at index j, and moreover, $b_j = C(x_j, w_j)$.
  - **Gate computation:** For every gate g in P with input wires $j_{\mathrm{L}}, j_{\mathrm{R}}$ and output wire j, it proves that there exists a local opening of com to wire values $b_{j_{\mathrm{L}}}, b_{j_{\mathrm{R}}}, b_i \in \{0,1\}$ at indices $j_{\mathrm{L}}, j_{\mathrm{R}}, j \in [s]$, respectively, and moreover, $b_j = g(b_{j_{\mathrm{L}}}, b_{j_{\mathrm{R}}})$.
  - **Output wire**: It proves that there exists a local opening to the value 1 at index s for com.

  The proof consists of the commitment com together with the batch argument π.

When the policy circuit P has logarithmic depth, the authors of [BBK+23] describe a simple inductive argument to argue the security of this construction by relying on somewhere extractability of the underlying BARG. Somewhere extractability says that the common reference string of the BARG can be programmed at a small number of (hidden) indices $i_1, \ldots, i_\ell$. Given a valid proof π for $(x_1, \ldots, x_n)$ along with a trapdoor, one can extract witnesses for $x_{i_1}, \ldots, x_{i_\ell}$. However, when P has super-logarithmic depth, the basic inductive argument no longer suffices (specifically, the security loss of the reduction decays *exponentially* in the depth of P).

Predicate-Extractable Hash Functions for Bit-Fixing Constraints. To construct monotone policy BARGs for policies P of arbitrary depth, the authors of [BBK+23] replace the Merkle hash of the wire values with a more sophisticated "predicate-extractable" hash function for bit-fixing constraints.[1]

A predicate-extractable hash function for bit-fixing predicates is a hash function where the hash key can be programmed in one of two computationally indistinguishable modes: (1) a normal mode and (2) a bit-fixing mode. In bit-fixing mode, the setup algorithm takes as input a set of indices $S \subseteq [n]$ along with a collection of bits $\{(i, y_i)\}_{i \in S}$, where n is the input length. It outputs a hash key hk and an extraction trapdoor td. The correctness requirement says that if $\mathsf{dig} = \mathsf{Hash}(\mathsf{hk}, \mathbf{x})$ for an input $\mathbf{x}$ where $x_i = y_i$ for all $i \in S$, then $\mathsf{Extract}(\mathsf{td}, \mathsf{dig}) = \mathsf{Matching}$. Alternatively, if dig is a digest for an input $\mathbf{x}$ where $x_i \neq y_i$ for some $i \in S$, then $\mathsf{Extract}(\mathsf{td}, \mathsf{dig})$ should output $(\mathsf{NotMatching}, i^*)$ where $i^* \in S$ is an index where $x_{i^*} \neq y_{i^*}$. Essentially, the extractor is deciding whether dig corresponds to the hash of an input that is consistent with

[1] This is conceptually similar to the notion of function-binding hash functions introduced concurrently in [FWW23].

$\{(i, y_i)\}_{i \in S}$. If the hash is declared inconsistent, the extractor outputs one of the inconsistent indices. Finally, the hash function supports succinct local openings to individual bits of an input. The two key security properties are as follows:

- For a hash digest dig where $\mathsf{Extract}(\mathsf{td}, \mathsf{dig}) = \mathsf{Matching}$, then it should be computationally difficult to construct an opening for dig to a value $x_i \neq y_i$ for any $i \in S$.
- For a hash digest dig where $\mathsf{Extract}(\mathsf{td}, \mathsf{dig}) = (\mathsf{NotMatching}, i^*)$, then it should be computationally difficult for the adversary to open index i^* to the value y_{i^*}.

In the monotone BARG construction, the prover takes the Boolean circuit C, the policy P, the statements $(x_1, \ldots, x_k)$ and the witnesses $(w_1, \ldots, w_k)$, and computes $b_i \leftarrow C(x_i, w_i)$ and $P(b_1, \ldots, b_k)$. Let $(b_1, \ldots, b_s)$ be the complete set of wire values in $P(b_1, \ldots, b_k)$, arranged in topological order. The prover hashes the wire values $(b_1, \ldots, b_s)$ using the predicate-extractable hash function. In fact, the prover computes two independent hashes $\mathsf{dig}_1, \mathsf{dig}_2$ of the wire values, and the BARG will check validity of the openings against *both* hashes. To argue non-adaptive soundness, the authors of [BBK+23] first define the zero-set J associated with a circuit C, policy P, and statement $(x_1, \ldots, x_k)$:

- For each $i \in [k]$, let $\beta_i^* = 1$ if there exists w_i such that $C(x_i, w_i) = 1$ and let $\beta_i^* = 0$ otherwise.
- Let $\beta_1^*, \ldots, \beta_s^* = P(\beta_1^*, \ldots, \beta_k^*)$ be the wire values in $P(\beta_1^*, \ldots, \beta_s^*)$, where the wires are ordered topologically.
- Let $J = \{i \in [k] : \beta_i^* = 0\}$. For a layer index t, define $J_t \subseteq J$ to just contain the indices of wires in layer t of P.

The proof of non-adaptive soundness now proceeds as follows:[2]

- Take any circuit C, monotone policy P, and statements $x_1, \ldots, x_k$. The invariant they use roughly says the following: if $\mathsf{hk}_0, \mathsf{hk}_1$ are programmed to bind to the all-zeroes string on the zero-sets J_i, J_{i-1} for layers i and $i-1$ of P, and the digest associated with the upper layer is $\mathsf{NotMatching}$, then the digest associated with the lower layer is also $\mathsf{NotMatching}$.
- To establish this invariant, the proof critically relies on BARG security and security of the predicate-extractable hash function. Namely, if the extractor declares an index $j \in J_i$ in the upper layer to be $\mathsf{NotMatching}$ and the BARG is set to be extracting on wire j, then that means the adversary must have opened one of the input wires j' (to the gate computing wire j) to a 1 where $j' \in J_{i-1}$ (since the policy P is monotone). Security of the hash function then says that the extractor must declare the digest associated with the lower layer to be $\mathsf{NotMatching}$.

[2] With a suitable strengthening of the notion of predicate-extractable hash functions, the authors of [BBK+23] also show how to obtain a somewhere *extractable* monotone policy BARG. In this work, we focus on achieving the core notion of non-adaptive soundness.

– To complete the proof, they argue that the output layer must be NotMatching (by programming the BARG to be extracting on the output wire). By propagating the invariant to the input wires, they conclude that the input layer must be NotMatching (when one of the hash keys is programmed to bind on the input layer). In this case, programming the BARG to be extracting on the wire identified by the NotMatching input (output by the extractor for the hash function) yields a contradiction (in this case, the BARG extractor would need to output a witness for a false NP statement).

The authors of [BBK+23] then show how to construct a predicate-extractable hash function for bit-fixing predicates using the learning with errors (LWE) assumption. Their construction specifically relies on leveled homomorphic encryption (similar to the construction of somewhere statistically binding hash functions [HW15]). In conjunction with BARGs for NP based on LWE [CJJ21b], this yields a monotone policy BARG for NP from LWE.

This Work: Zero-Fixing Hash Functions. The starting point of our work is a relaxation of a predicate-extractable hash function for bit-fixing predicates we call a zero-fixing hash function. Like the predicate-extractable hash function, the zero-fixing hash function supports succinct local openings and moreover, the hash key for a zero-fixing hash function can be sampled in one of two computationally-indistinguishable modes: (1) a normal mode and (2) a zero-fixing mode. In zero-fixing mode, the setup algorithm takes as input a set $S \subseteq [n]$ of indices (that should be zero) and outputs a hash key hk along with a trapdoor td. There is also an extract algorithm Extract that takes as input the hash key hk and a digest dig, and outputs either Matching or NotMatching. The key distinction with predicate-extractable hash functions is that Extract only outputs the flag; it does *not* output an index when it declares a digest NotMatching. Correspondingly, the zero-fixing security requirement only imposes a requirement for matching digests:

– **Zero-fixing:** Suppose $(\mathsf{hk}, \mathsf{td})$ are sampled in zero-fixing mode for a set S. Then, for any digest dig where $\mathsf{Extract}(\mathsf{td}, \mathsf{dig})$ outputs Matching, it should be hard to find an opening to an index $i \in S$ to the value 1.

While this distinction of having the extractor output a mismatching index j or not might seem like a small difference, it has two significant implications:

– **Simpler to construct:** By only requiring the zero-fixing hash function declare whether a digest is Matching or NotMatching, we significantly simplify the construction of the hash function. Whereas computing and propagating an index of a "mismatching bit" (as in [BBK+23]) relies heavily on (leveled) homomorphic encryption, checking whether there *exists* a mismatching index or not can be realized from simpler tools. As we show in this work (and describe later on), we can construct zero-fixing hash functions generically from BARGs for NP together with any *additively* homomorphic encryption scheme (Section 5). If we prefer to avoid non-black-box techniques altogether, we also describe a direct algebraic construction using composite-order pairing

groups (the full version [NWW23, Section 6]). This is the critical distinction that allows us to obtain monotone policy BARGs from group-based assumptions (which give additively homomorphic encryption [Gam84] but not leveled homomorphic encryption).

– **Sufficient for monotone policy BARGs:** A second important fact is that our notion of zero-fixing hash function still suffices to build monotone policy BARGs. As noted in the preceding sketch, the soundness analysis from [BBK+23] critically relied on the hash function extractor outputting an index of a mismatching bit. This is so that when the BARG is programmed to bind on the wire associated with the mismatching index, the NotMatching invariant propagates from the output layer to the input layer. In our setting, the zero-fixing extractor only outputs Matching or NotMatching, and in the case where the extractor outputs NotMatching, we *cannot* definitively declare an index to be "mismatching." This requires a new proof strategy as well as imposing additional security requirements on the zero-fixing hash function. We describe these properties as well as our new proof strategy in more detail below.

Monotone Policy BARGs from Zero-Fixing Hash Functions. Our main construction is similar to the canonical protocol from [BBK+23] sketched above, except the prover commits to all of the wires of the policy circuit P using two zero-fixing hash functions (with hash keys hk_1 and hk_2). Our security analysis takes a different *bottom-up* approach rather than the previous top-down approach. The bottom-up approach is more natural when using our zero-fixing hash function. Here, we provide a sketch of our non-adaptive soundness analysis.

To argue non-adaptive soundness, fix a Boolean circuit C, a monotone policy P (assumed to be a layered Boolean circuit), and a false statement $(x_1, \ldots, x_k)$. Similar to [BBK+23], we define the zero-set J associated with C, P, and $(x_1, \ldots, x_k)$. The zero-set J contains the indices of the wires with value 0 in the computation $P(\beta_1^*, \ldots, \beta_k^*)$ where $\beta_i^* = 1$ if there exists w_i where $C(x_i, w_i) = 1$ and 0 otherwise. Since P is monotone, for all $w_1, \ldots, w_k$, the wire values of $P(C(x_1, w_1), \ldots, C(x_k, w_k))$ on the set J will be zero. As before, let $J_i \subset J$ be the subset of wires in layer i of P.

Our soundness argument proceeds layer-by-layer, starting from the input layer (i.e., layer 1) and progressing to the output layer (i.e., layer d, where d is the depth of P). Our goal establishes the following invariant: if the hash keys hk_1 and hk_2 are zero-fixing on J_i and J_{i+1} and the digest associated with the lower layer (i.e., layer i) is Matching, then the digest associated with the upper layer (i.e., layer $i+1$) is also Matching. We provide a sketch of this step. For ease of exposition, suppose hk_1 is zero-fixing on J_i and the digest dig_1 is Matching.

The goal is to show that hk_2 is zero-fixing on J_{i+1}, then the digest dig_2 is also Matching:[3]

- Initially, we set hk_2 to be binding on the empty set. We require in this case that dig_2 is always Matching. We refer to this property as an extractor validity property on the zero-fixing hash function.
- We now iteratively build up hk_2. Let $J_{i+1}[1]$ be the first element of J_{i+1}. We set hk_2 to be binding on the set $\{J_{i+1}[1]\}$. Our goal is to argue that dig_2 is still Matching. While it might seem like this property should follow assuming a basic index hiding property on the zero-fixing hash function (i.e., that the hash key hk hides which set it is binding on), this is *insufficient*. The reason is that when hk_2 is binding on $\varnothing$, the adversary might output a Matching digest dig_2, but if hk_2 is binding on $\{J_{i+1}[1]\}$, the output digest dig_2 might be NotMatching. We cannot use such an adversary to construct an index hiding distinguisher, because in the index hiding security game, the distinguisher does *not* have the extraction trapdoor. As such, an attempted reduction algorithm cannot efficiently decide whether the adversary was successful or not. Indeed, this is a fundamental issue since knowledge of the extraction trapdoor would trivially break index hiding.
- To advance the proof, we introduce a *stronger* notion of index hiding security for zero-fixing hash functions, which essentially requires that no efficient adversary can output a digest dig that causes the output of Extract to differ depending on whether the hash key is binding on a set S or a set $S \setminus \{i\}$.[4] Of course, this is only meaningful when the digest is computed over an input that is 0 on index i.[5] Thus, we require this stronger index hiding with extracted guess property to hold only for digests dig where the adversary can provide an opening to index 0 for the target index i. We define this property formally in Definition 6.
- To leverage the index hiding with extracted guess property, we need to enforce the fact that dig_2 opens to a 0 on index $J_{i+1}[1]$. We ensure this by appealing to the somewhere extractability of the BARG along with zero-fixing security of the hash function. Specifically, suppose that the BARG is binding on wire

[3] This step is straightforward if we had a predicate-extractable hash function where the extractor outputs a mismatching index. Namely, if the upper layer digest is NotMatching, then the extractor outputs an index $j \in J_{i+1}$ that is mismatching (i.e., cannot be opened to a 0). This means the efficient adversary can only open wire j to the value 1. Now, if the BARG is extracting on the statement associated with wire j, then we either (1) obtain the opening of some index $j' \in J_i$ to a 1, which breaks security of the hash function (since the lower layer digest is Matching); or (2) the value of wire j is inconsistent with the input wires associated with the gate computing wire j, which breaks security of the BARG.

[4] This type of property where the output of the extractor does not change for different choices of the CRS is often referred to as a "no-signaling" extraction property [PR17, KPY19, GZ21, KVZ21, CJJ21b].

[5] Otherwise, an honest digest on the input that is 1 at index i (and 0 everywhere else) would be declared Matching if the hash key was zero-fixing on a set S that contains i and NotMatching if the hash key was zero-fixing on the set $S \setminus \{i\}$.

$J_{i+1}[1]$. The BARG extractor then produces openings to the wire $J_{i+1}[1]$ with respect to dig_2 as well as opening to the wires $j_{\mathrm{L}}, j_{\mathrm{R}}$ with respect to dig_1 (corresponding to the input wires for the gate computing $J_{i+1}[1]$). Since dig_1 is zero-fixing on J_i and dig_1 is also $\mathsf{Matching}$, if either $j_{\mathrm{L}}, j_{\mathrm{R}} \in J_i$, then the extracted openings must be openings to 0 (otherwise, we break zero-fixing of the hash function). But by monotonicity of P, this means the value of the output wire $J_{i+1}[1]$ must also be 0, and thus the BARG extractor produces an opening to 0 for wire $J_{i+1}[1]$. Now, by the index hiding with extracted guess property, we conclude that programming hk_2 to zero-fix on set $\{J_{i+1}[1]\}$ will still cause dig_2 to be $\mathsf{Matching}$ (except with a negligible loss in probability).
- We can now iteratively apply the argument and build up hk_2 until it is binding on all of J_{i+1}.

To complete the proof, we consider the input and output layers for P:

- **Handling the input layers:** The base case in our analysis is to show that if hk_1 is binding on J_1 (the input layer), then it is $\mathsf{Matching}$. This follows using the same layer-wise strategy sketched above for proving our invariance, except for each index $J_1[i]$, we rely on the fact that the associated statement x_i is false (i.e., no witness exists) to argue that the only valid opening for dig_1 on index i is 0. Otherwise, we either break somewhere extractability of the BARG (i.e., extracting an invalid witness for index i) or the index hiding with extracted guess property.
- **Output layer:** Starting from the input layer, we now iteratively apply our basic invariant to argue that when the hash keys are binding to J_d (the output layer), the associated digests are also $\mathsf{Matching}$. Now, if we have a valid proof, and the BARG is set to extract on the output layer, then the BARG extractor outputs an opening of the output wire to 1 with respect to the hash digests. However, since the output wire is contained in J_d (since the statement is false), and the digest is matching, this breaks zero-fixing security of the hash function.

Thus, the above analysis suffices to show non-adaptive soundness of our construction. The critical security requirement we require on our zero-fixing hash function is the strengthened index hiding with extracted guess property. This property allows us to complete the proof via an iterative approach without needing to rely on the extractor outputting a mismatching index as in previous work [BBK+23]. As we discuss below, this is an easier property to realize than full-fledged index extraction. We refer to Section 3 for the formal definition of zero-fixing hash functions and Section 4 for our construction of monotone policy BARGs.

Constructing Zero-Fixing Hash Functions. Our second contribution in this work is a generic construction of zero-fixing hash functions from vanilla BARGs together with an additively homomorphic encryption scheme. We start with a basic construction that captures the key ideas underlying our construction and refer to Section 5 for the formal description and analysis:

- Let $n \in \mathbb{N}$ be the input length. For ease of exposition, we assume that $n = 2^k$ is a power-of-two. Suppose we want to zero-fix on a (possibly-empty) set $S \subseteq [n]$. The setup algorithm first samples a public/secret key-pair $(\mathsf{pk}, \mathsf{sk})$ for an additively homomorphic encryption scheme. For each $i \in [n]$, the setup algorithm construct an encryption $\mathsf{ct}_i \leftarrow \mathsf{Enc}(\mathsf{pk}, 1)$ of 1 if $i \in S$ and an encryption of $\mathsf{ct}_i \leftarrow \mathsf{Enc}(\mathsf{pk}, 0)$ of 0 if $i \notin S$. It also constructs an encryption $\mathsf{ct}_{\mathsf{zero}} \leftarrow \mathsf{Enc}(\mathsf{pk}, 0)$ of 0. Finally, it constructs a commitment $\mathsf{com}_{\mathsf{hk}}$ to the ciphertexts $(\mathsf{ct}_1, \ldots, \mathsf{ct}_n)$. The hash key is then $\mathsf{hk} = (\mathsf{pk}, \mathsf{ct}_{\mathsf{zero}}, \mathsf{ct}_1, \ldots, \mathsf{ct}_n, \mathsf{com}_{\mathsf{hk}})$, and the extraction trapdoor is the decryption key sk.
- To hash an input $x \in \{0,1\}^n$, the user constructs a complete binary tree where each of the n leaves is associated with a ciphertext. If $x_i = 1$, then the user associates leaf i with ct_i, and if $x_i = 0$, then the user associates leaf i with $\mathsf{ct}_{\mathsf{zero}}$. The value of each internal node in the binary tree is defined to be the sum of the ciphertexts associated with its two children. By construction, the value of the root node is an encryption of the sum of the values associated with the n leaf nodes. We refer to the tree of ciphertexts as the "ciphertext-evaluation tree." The digest dig then consists of the ciphertext $\mathsf{ct}_{\mathsf{root}}$ associated with the root node along with a commitment $\mathsf{com}_{\mathsf{ct}}$ to all of the ciphertexts in the ciphertext-evaluation tree.
- A local opening for index i^* and value $b_{i^*} \in \{0,1\}$ for the digest $\mathsf{dig} = (\mathsf{ct}_{\mathsf{root}}, \mathsf{com}_{\mathsf{ct}})$ is a BARG proof. The BARG statements correspond to the indices of the nodes in the ciphertext-evaluation tree. The associated relation is parameterized by the target index i^*, the root ciphertext $\mathsf{ct}_{\mathsf{root}}$, the encryption $\mathsf{ct}_{\mathsf{zero}}$ of 0 from the hash key, and the commitment to the input ciphertexts $\mathsf{com}_{\mathsf{hk}}$. The BARG relation then checks the following:
  - **Leaf nodes:** For each leaf node i, $\mathsf{com}_{\mathsf{ct}}$ opens to either $\mathsf{ct}_{\mathsf{zero}}$ or ct_i at index i. For the particular index i^*, it checks that $\mathsf{com}_{\mathsf{ct}}$ opens to $\mathsf{ct}_{\mathsf{zero}}$ if $b_{i^*} = 0$ and $\mathsf{com}_{\mathsf{ct}}$ opens to ct_{i^*} if $b_{i^*} = 1$. Since the BARG relation only has $\mathsf{com}_{\mathsf{hk}}$ and not ct_i itself, the prover provides ct_i as part of its witness along with a proof of opening for ct_i with respect to $\mathsf{com}_{\mathsf{hk}}$. The proof of opening ensures that the correct ct_i is provided.
  - **Internal nodes:** For an internal node i (with children indexed $j_{\mathrm{L}}, j_{\mathrm{R}}$), the BARG checks that $\mathsf{com}_{\mathsf{ct}}$ opens to ciphertexts ct_i, $\mathsf{ct}_{j_{\mathrm{L}}}$, $\mathsf{ct}_{j_{\mathrm{R}}}$ where ct_i is the sum of ciphertexts $\mathsf{ct}_{j_{\mathrm{L}}}$ and $\mathsf{ct}_{j_{\mathrm{R}}}$.
  - **Root node:** For the root node, the BARG checks that $\mathsf{com}_{\mathsf{ct}}$ opens to $\mathsf{ct}_{\mathsf{root}}$.
- To test whether a digest $\mathsf{dig} = (\mathsf{ct}_{\mathsf{root}}, \mathsf{com}_{\mathsf{ct}})$ is matching or not, the $\mathsf{Extract}$ algorithm outputs $\mathsf{Matching}$ if $\mathsf{ct}_{\mathsf{root}}$ decrypts to 0 and $\mathsf{NotMatching}$ otherwise.

By definition, the ciphertext $\mathsf{ct}_{\mathsf{root}}$ in any (honestly-generated) hash digest is the sum of the ciphertexts associated with the leaves of the ciphertext-evaluation tree. On an input x, if $x_i = 0$, then the associated ciphertext is an encryption of 0 and does not contribute to the sum. If $x_i = 1$, then the ciphertext associated with the leaf is an encryption of 1 if $i \in S$ and encryption of 0 otherwise. Thus, the sum is only incremented if $x_i = 1$ for some $i \in S$. This is precisely when

$\mathsf{Extract}$ outputs $\mathsf{NotMatching}$ (i.e., the digest is for an input x where $x_i = 1$ for $i \in S$).

To argue that it is hard to open a $\mathsf{Matching}$, but possibly-malformed digest to a 1 at an index $i \in S$, we appeal to soundness of the BARG. In this case, the root ciphertext $\mathsf{ct}_{\mathsf{root}}$ in dig decrypts to a non-zero value, and yet the user constructed a valid BARG proof of opening for an index $i \in S$. The key observation is that the structure of the BARG used in the above construction is very similar to the structure of the canonical protocol from [BBK+23] described at the beginning of Section 1.2 for demonstrating correct evaluation of a monotone circuit. Moreover, because the ciphertext-evaluation tree is *perfectly* balanced, it has depth $\log n$, where $n = \mathsf{poly}(\lambda)$ is the input length. As such, we are able to adapt the proof strategy for arguing soundness of the monotone policy BARGs for *log-depth circuits* to directly argue zero-fixing security of our hash function. Specifically, we rely on BARG security to ensure that if the adversary uses an encryption of 1 as one of the leaves to the ciphertext (which it must if it opens an index $i \in S$ to a 1), then the root ciphertext necessarily is an encryption of a non-zero value. We provide the full details in the full version [NWW23, Section 5].

While the core construction described here satisfies zero-fixing security, we need to augment the construction to satisfy the additional security requirements we impose on a zero-fixing hash function. We summarize these here, and defer to the technical sections (see Section 5 and the full version of this paper [NWW23]) for the full details:

- **Extractor validity:** Recall that this property says that when the hash function is zero-fixing on the empty set, it should be hard for an adversary to come up with a "valid" digest that is $\mathsf{NotMatching}$. To satisfy this property, we simply include a BARG proof of validity to the digest, where the BARG proof of validity simply checks that the ciphertext-evaluation tree was correctly constructed. When the hash key is binding to the empty set, all of the ciphertexts ct_i are an encryption of 0, so the root of a properly computed ciphertext-evaluation tree will also be an encryption of 0. We provide the details in the full version [NWW23, Section 5].
- **Index hiding with extracted guess:** Recall that this property says that the adversary cannot produce a digest dig where the extractor output disagrees depending on whether the hash key is zero-fixing on a set S or a set $S \setminus \{i\}$ (provided that the adversary provides an opening to 0 for index i). The only difference between the hash keys in these two cases is ct_i in the CRS changes from an encryption of 0 to an encryption of 1, which we could in principle show using semantic security. However, the reduction algorithm would have no way of checking whether a digest dig output by the adversary is $\mathsf{Matching}$ or $\mathsf{NotMatching}$ (since it does *not* and cannot know the decryption key). Thus, to argue this we adopt a Naor-Yung type of strategy [NY90] and *encrypt* twice. Namely, we introduce two *parallel* copies of the scheme (i.e., two independent public keys and two independent sets of ciphertexts). The digest now consists of two ciphertexts $\mathsf{ct}^{(0)}_{\mathsf{root}}, \mathsf{ct}^{(1)}_{\mathsf{root}}$ for the roots of the two ciphertext-evaluation trees. The same BARG would validate both roots. The key idea now is we can

switch $\mathsf{ct}_i^{(0)}$ from an encryption of 0 (i.e., zero-fixing at $S\backslash\{i\}$) to an encryption of 1 (i.e., zero-fixing at S) while being able to decrypt (i.e., extract) for the parallel encryption scheme. We can leverage soundness of the BARG to argue that for a valid digest/opening, both $\mathsf{ct}_{\mathsf{root}}^{(0)}$ and $\mathsf{ct}_{\mathsf{root}}^{(1)}$ encrypt *identical* values. This allows us to leverage semantic security to switch the ciphertexts for one scheme while being able to detect whether the output of $\mathsf{Extract}$ changed or not (using knowledge of the secret key for the parallel scheme). We provide the full details in the full version [NWW23, Section 5].

Taken together, we obtain a zero-fixing hash function from any standard BARG together with an additively-homomorphic encryption scheme. By instantiating with BARGs from the k-Lin assumption over pairing groups [WW22] or the (sub-exponential) DDH assumption over pairing-free groups [CGJ+23], we obtain zero-fixing hash functions from the same underlying assumptions. In conjunction with our generic construction from above, this yields Theorem 1.

An Algebraic Construction of Zero-Fixing Hash Functions. As another contribution, we also describe an algebraic approach to construct zero-fixing hash functions directly from (composite-order) bilinear maps. This construction has the advantage that it only makes black-box use of cryptography. We give a brief sketch of the construction here, but defer the details to the full version [NWW23, Section 6]. The basic version is an adaptation of the Catalano-Fiore vector commitment [CF13]:

- Let $\mathcal{G} = (\mathbb{G}, \mathbb{G}_T, N, g, e)$ be a composite-order bilinear group of order N, generator g, and an efficiently-computable non-degenerate bilinear map $e\colon \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. In the actual construction, we will require that N be a product of six primes. In the description here, we will just describe the basic scheme that operates primarily in just two subgroups. Let g_1 and g_2 be generators of two orthogonal subgroups of $\mathbb{G}$.
- To sample a hash key for a set $S \subseteq [n]$, the setup algorithm samples exponents $\alpha_i, \beta_i \xleftarrow{R} \mathbb{Z}_N$. If $i \in S$, it sets $A_i \leftarrow (g_1 g_2)^{\alpha_i}$ and if $i \notin S$, it sets $A_i \leftarrow g_1^{\alpha_i}$. It sets $B_i \leftarrow g_1^{\beta_i}$ and for $i \neq j$, it computes the cross term $C_{i,j} \leftarrow g_1^{\alpha_i \beta_j}$. The hash key then contains A_i, B_i for $i \in [n]$ and $C_{i,j}$ for all $i \neq j$.
- The hash of an input $x \in \{0,1\}^n$ is then $\mathsf{dig} = \prod_{i \in [n]} A_i^{x_i}$. The opening to an index i is $V = \prod_{j \neq i} C_{j,i}^{x_j}$. To verify an opening to a bit b at index i, the verifier checks
$$e(\mathsf{dig}, B_i) = e(A_i, B_i)^b \cdot e(g_1, V).$$
- To check whether a digest dig is $\mathsf{Matching}$ or not, the extraction algorithm output $\mathsf{Matching}$ if $e(\mathsf{dig}, g_2) = 1$ and $\mathsf{NotMatching}$ otherwise.

The basic principle is to move the "encoding elements" A_i for $i \in S$ to have a component in the span of g_2. The components A_i for $i \notin S$ are only in the span of g_1. Then, any digest that includes an index $i \in S$ will contain a non-zero element in the span of g_2, and thus, be declared $\mathsf{NotMatching}$. Arguing the security of this scheme is more delicate and will require introducing a number of additional

randomizing components (and subgroups). We refer to the full version [NWW23, Section 6] for the details.

Constructing Monotone Policy Aggregate Multi-Signatures. Our final contribution is a construction of monotone policy aggregate multi-signatures. While previous construction of aggregate signatures relied on extractable BARGs [WW22,DGKV22], a similar implication is possible by combining a non-adaptively-sound BARG together with a "puncturable signature" scheme (also called an all-but-one signature scheme) [GVW19]. We sketch our construction below, and provide the full details in the full version [NWW23, Section 7].

In a puncturable signature scheme, it is possible to puncture a verification key on a message m^*. The property is that there does not *exist* signatures on m^* that verify with respect to the punctured verification key. Moreover, a punctured verification key is computationally indistinguishable from an honestly-generated verification key, even if the adversary is able to see signatures on arbitrary messages $m \neq m^*$. Goyal, Vusirikala, and Waters [GVW19] showed how to construct puncturable signatures from most standard number-theoretic assumptions (e.g., RSA, pairing-based assumptions, and LWE). We can use a non-adaptively-sound monotone policy BARG together with a puncturable signature scheme to construct a (statically-secure)[6] aggregate multi-signature scheme for any policy computed by a monotone Boolean circuit. We provide a sketch below:

- **Setup:** Consider a scheme with k signers. Each signer $i \in [k]$ has a signing key sk_i and a verification key vk_i for the punctured signature scheme. The public parameters of the aggregation scheme contain the common reference string for a monotone policy BARG.
- **Signing:** To sign a message m, each user signs with their individual signing key.
- **Aggregation:** Given a set of signatures $\{\sigma_i\}_{i \in S}$ on the same message m and a (monotone) aggregation policy P, a user can aggregate the signatures by giving a monotone policy BARG proof for the policy P with respect to the natural relation $\mathcal{R}[m] = \{(\mathsf{vk}, \sigma) : \mathsf{Verify}(\mathsf{vk}, m, \sigma)\}$. The aggregate signature is simply the BARG proof for the statements $(\mathsf{vk}_1, \ldots, \mathsf{vk}_k)$ with the witness $(\sigma_1, \ldots, \sigma_k)$.
- **Verification:** To verify an aggregate multi-signature with respect to a policy P, the verifier just checks the BARG proof.

Note that one could also construct an aggregate multi-signature by sending the set S where $P(S) = 1$ and then use a vanilla BARG to prove knowledge of a signature σ_i for every $i \in S$. However, this approach would require communicating the set S as part of the aggregate signature. Using monotone policy BARGs, the aggregate signature only consists of the BARG proof, and thus has size, $\mathsf{poly}(\lambda, \log |P|)$. It is straightforward to prove static security of the above

[6] In the static security model, we require that the adversary declare the set of corrupted verification keys, its challenge message, and the aggregation policy at the beginning of the security game.

multi-signature scheme just assuming *non-adaptive-soundness* on the underlying BARG. We sketch the reduction below:

- In the static security game, the adversary has to pre-commit to the message m^* it wants to forge on, the set of verification keys $(\mathsf{vk}_1^*, \ldots, \mathsf{vk}_k^*)$ it wants to use (which can be a mix of honest verification keys chosen by the challenger and verification keys chosen adversarially), and the aggregation policy P *before* seeing the aggregation parameters.
- Let $S \subseteq [k]$ be the set of indices i where the chosen key vk_i^* is uncorrupted (i.e., chosen by the challenger). The admissibility requirement is that $P(b_1, \ldots, b_k) = 0$ where $b_i = 0$ if $i \in S$ and $b_i = 1$ otherwise; this is saying that the adversary cannot satisfy the policy P just by providing signatures under keys it controls.
- In the security reduction, we first puncture the honest users' verification keys vk_i on the challenge message m^*. This means that there does not exist valid signatures on the challenge message m^* with respect to the honest users' verification keys vk_i
- Consider the relation $\mathcal{R}[m^*]$ used for verification. By definition of the set S and the fact that the honest verification keys are punctured at m^*, the statement $(\mathsf{vk}_1^*, \ldots, \mathsf{vk}_k^*)$ is *false* for the policy P with respect to the relation $\mathcal{R}[m^*]$. By non-adaptive soundness of the monotone policy BARG, the probability that the adversary can produce a valid aggregate signature (i.e., a valid proof on a false statement) is negligible.

Observe that in the above sketch, the verification time is linear in k. However, using a RAM delegation scheme, we can achieve fast verification. We refer to the full version [NWW23, Remark 7.8] for additional details.

2 Preliminaries

Throughout this work, we write λ to denote the security parameter. For $n \in \mathbb{N}$, we write $[n]$ to denote the set $\{1, \ldots, n\}$. For $a, b \in \mathbb{N}$ we write $[a, b]$ to denote the set $\{a, a+1, \ldots, b\}$. We write $\mathsf{poly}(\lambda)$ to denote a function that is bounded by a fixed polynomial in λ, and $\mathsf{negl}(\lambda)$ to denote a function that is $o(\lambda^{-c})$ for all $c \in \mathbb{N}$. For a finite set S, we write $x \xleftarrow{R} S$ to denote that x is a uniformly random element of S. For a distribution $\mathcal{D}$ we write $x \leftarrow \mathcal{D}$ to denote that x is a random draw from $\mathcal{D}$.

We say an algorithm is efficient if it runs in probabilistic polynomial time in the length of its input. A non-uniform algorithm $\mathcal{A}$ consists of a pair of algorithms $(\mathcal{A}_1, \mathcal{A}_2)$ where $\mathcal{A}_1$ is a (possibly-unbounded) algorithm that takes as input 1^λ and outputs an advice string ρ_λ of $\mathsf{poly}(\lambda)$ size. Algorithm $\mathcal{A}_2$ is an efficient algorithm. The output of $\mathcal{A}$ on an input $x \in \{0,1\}^\lambda$ is defined as first computing the advice string $\rho_\lambda \leftarrow \mathcal{A}_1(1^\lambda)$ and then outputting $\mathcal{A}_2(x, \rho_\lambda)$. We say two ensembles of distributions $\mathcal{D}_1 = \{\mathcal{D}_{1,\lambda}\}_{\lambda \in \mathbb{N}}$ and $\mathcal{D}_2 = \{\mathcal{D}_{2,\lambda}\}_{\lambda \in \mathbb{N}}$ are computationally indistinguishable if no efficient adversary can distinguish them with non-negligible probability. We say they are statistically indistinguishable if their statistical distance is bounded by $\mathsf{negl}(\lambda)$.

2.1 Batch Arguments for NP

In this section, we recall the notion of a non-interactive batch argument (BARG) for NP, the special case of a BARG for index languages and the notion of a BARG for monotone policy batch NP [BBK+23].

Batch Arguments for NP. We begin with the notion of a somewhere extractable batch argument for NP. Our presentation is adapted from [CJJ21b, WW22]. Here, we provide a more general syntax where the batch arguments supports extraction on up to ℓ indices.

Definition 1 (Boolean Circuit Satisfiability). *We define the circuit satisfiability language* $\mathcal{L}_{\mathsf{CSAT}}$ *as*

$$\mathcal{L}_{\mathsf{CSAT}} = \left\{ (C, x) \;\middle|\; \begin{array}{c} C\colon \{0,1\}^n \times \{0,1\}^h \to \{0,1\}, x \in \{0,1\}^n \\ \exists w \in \{0,1\}^* : C(x,w) = 1 \end{array} \right\}.$$

Definition 2 (Non-Interactive Batch Argument). *A somewhere extractable non-interactive batch argument (BARG) for Boolean circuit satisfiability is a tuple of efficient algorithms* $\Pi_{\mathsf{BARG}} = (\mathsf{Gen}, \mathsf{Prove}, \mathsf{Verify}, \mathsf{TrapGen}, \mathsf{Extract})$ *with the following syntax:*

- $\mathsf{Gen}(1^\lambda, 1^k, 1^n, 1^s, 1^\ell) \to (\mathsf{crs}, \mathsf{vk})$*: On input the security parameter* $\lambda \in \mathbb{N}$*, the number of instances* $k \in \mathbb{N}$*, the instance length* $n \in \mathbb{N}$*, a bound on the size of the Boolean circuit* $s \in \mathbb{N}$*, and a bound on the size of the extraction set* $\ell \in \mathbb{N}$*, the generator algorithm outputs a common reference string* crs *and a verification key* vk*.*
- $\mathsf{Prove}(\mathsf{crs}, C, (x_1, \ldots, x_k), (w_1, \ldots, w_k)) \to \pi$*: On input the common reference string* crs*, a Boolean circuit* $C\colon \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$*, statements* $x_1, \ldots, x_k \in \{0,1\}^k$*, and witnesses* $w_1, \ldots, w_k \in \{0,1\}^h$*, the prove algorithm outputs a proof* π*.*
- $\mathsf{Verify}(\mathsf{vk}, C, (x_1, \ldots, x_k), \pi) \to b$*: On input the verification key* vk*, a Boolean circuit* $C\colon \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$*, statements* $x_1, \ldots, x_k \in \{0,1\}^n$ *and a proof* π*, the verification algorithm outputs a bit* $b \in \{0,1\}$*.*
- $\mathsf{TrapGen}(1^\lambda, 1^k, 1^n, 1^s, 1^\ell, S) \to (\mathsf{crs}, \mathsf{vk}, \mathsf{td})$*: On input the security parameter* $\lambda \in \mathbb{N}$*, the number of instances* $k \in \mathbb{N}$*, the instance size* $n \in \mathbb{N}$*, a bound on the size of the Boolean circuit* $s \in \mathbb{N}$*, a bound on the size of the extraction set* $\ell \in \mathbb{N}$*, and a set* $S \subseteq [k]$ *of size at most* ℓ*, the trapdoor generator algorithm outputs a common reference string* crs*, a verification key* vk *and an extraction trapdoor* td*.*
- $\mathsf{Extract}(\mathsf{td}, C, (x_1, \ldots, x_k), \pi, i) \to w$*. On input the trapdoor* td*, a Boolean circuit* $C\colon \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$*, a collection of statements* $x_1, \ldots, x_k \in \{0,1\}^n$*, a proof* π *and an index* $i \in [k]$*, the extraction algorithm outputs a witness* w*.*

For notational convenience, when $\ell = 1$*, we omit the final input* 1^ℓ *and instead, write* $\mathsf{Gen}(1^\lambda, 1^k, 1^n, 1^s)$ *to denote* $\mathsf{Gen}(1^\lambda, 1^k, 1^n, 1^s, 1^1)$*. Similarly, we write* $\mathsf{TrapGen}(1^\lambda, 1^k, 1^n, 1^s, i)$ *to denote* $\mathsf{TrapGen}(1^\lambda, 1^k, 1^n, 1^s, 1^1, \{i\})$*. Finally, we require that* Π_{BARG} *satisfy the following properties:*

– ***Completeness:*** *For all* $\lambda, k, n, s, \ell \in \mathbb{N}$*, all Boolean circuits* $C\colon \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$ *of size at most* s*, all statements* $x = (x_1, \ldots, x_k) \in \{0,1\}^{kn}$ *and witnesses* $w = (w_1, \ldots, w_k) \in \{0,1\}^{kh}$ *where* $C(x_i, w_i) = 1$ *for all* $i \in [k]$*,*

$$\Pr\left[\mathsf{Verify}(\mathsf{vk}, C, (x_1, \ldots, x_k), \pi) = 1 : \begin{array}{c}(\mathsf{crs}, \mathsf{vk}) \leftarrow \mathsf{Gen}(1^\lambda, 1^k, 1^n, 1^s, 1^\ell) \\ \pi \leftarrow \mathsf{Prove}(\mathsf{crs}, C, x, w)\end{array}\right] = 1$$

– **Set hiding:** *For an adversary* $\mathcal{A}$ *and a bit* $b \in \{0,1\}$*, define the set hiding experiment* $\mathsf{ExptSH}_{\mathcal{A}}(\lambda, b)$ *as follows:*

1. *On input the security parameter* 1^λ*, algorithm* $\mathcal{A}$ *starts by outputting the number of instances* 1^k*, the instance size* 1^n*, the bound on the circuit size* 1^s*, the bound on the size of the extraction set* 1^ℓ*, and a set* $S \subseteq [k]$ *of size at most* ℓ*.*
2. *If* $b = 0$*, the challenger gives* $(\mathsf{crs}, \mathsf{vk}) \leftarrow \mathsf{Gen}(1^\lambda, 1^k, 1^n, 1^s, 1^\ell)$ *to* $\mathcal{A}$*. If* $b = 1$*, the challenger samples* $(\mathsf{crs}, \mathsf{vk}, \mathsf{td}) \leftarrow \mathsf{TrapGen}(1^\lambda, 1^k, 1^n, 1^s, 1^\ell, S)$ *and gives* $(\mathsf{crs}, \mathsf{vk})$ *to* $\mathcal{A}$*.*
3. *Algorithm* $\mathcal{A}$ *outputs a bit* $b' \in \{0,1\}$*, which is the output of the experiment.*

Then, Π_{BARG} *satisfies set hiding if for every efficient adversary* $\mathcal{A}$*, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that*

$$|\Pr[\mathsf{ExptSH}_{\mathcal{A}}(\lambda, 0) = 1] - \Pr[\mathsf{ExptSH}_{\mathcal{A}}(\lambda, 1) = 1]| = \mathsf{negl}(\lambda).$$

When $\ell = 1$*, we might refer to this property as* index hiding*.*

– **Somewhere extractable in trapdoor mode:** *For an adversary* $\mathcal{A}$*, define the somewhere extractable security game as follows:*

1. *On input the security parameter* 1^λ*, algorithm* $\mathcal{A}$ *starts by outputting the number of instances* 1^k*, the instance size* 1^n*, the bound on the circuit size* 1^s*, a bound on the size of the extraction set* 1^ℓ*, and a nonempty set* $S \subseteq [k]$ *of size at most* ℓ*.*
2. *The challenger samples* $(\mathsf{crs}, \mathsf{vk}, \mathsf{td}) \leftarrow \mathsf{TrapGen}(1^\lambda, 1^k, 1^n, 1^s, 1^\ell, S)$ *and gives* $(\mathsf{crs}, \mathsf{vk})$ *to* $\mathcal{A}$*.*
3. *Algorithm* $\mathcal{A}$ *outputs a Boolean circuit* $C\colon \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$ *of size at most* s*, statements* $x_1, \ldots, x_m \in \{0,1\}^n$*, and a proof* π*.*
4. *The output of the game is* $b = 1$ *if* $\mathsf{Verify}(\mathsf{vk}, C, (x_1, \ldots, x_m), \pi) = 1$ *and there exists an index* $i \in S$ *for which* $C(x_i, w_i) \neq 1$ *where* $w_i \leftarrow \mathsf{Extract}(\mathsf{td}, C, (x_1, \ldots, x_k), \pi, i)$*. Otherwise, the output is* $b = 0$*.*

Then Π_{BARG} *is somewhere extractable in trapdoor mode if for every adversary* $\mathcal{A}$*, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that* $\Pr[b = 1] = \mathsf{negl}(\lambda)$ *in the somewhere extractable game.*

– **Succinctness:** *There exists a fixed polynomial* $\mathsf{poly}(\cdot)$ *such that for all* $\lambda, k, n, s, \ell \in \mathbb{N}$*, all* crs *in the support of* $\mathsf{Gen}(1^\lambda, 1^k, 1^n, 1^s, 1^\ell)$*, and all Boolean circuits* $C\colon \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$ *of size at most* s*, the following properties hold:*

- **Succinct proofs:** *The proof* π *output by* $\mathsf{Prove}(\mathsf{crs}, C, \cdot, \cdot)$ *satisfies* $|\pi| \leq \mathsf{poly}(\lambda + \log k + s + \ell)$*.*
- **Succinct CRS:** $|\mathsf{crs}| \leq \mathsf{poly}(\lambda + k + n + \ell) + \mathsf{poly}(\lambda + \log k + s + \ell)$*.*

- **Succinct verification key:** $|\mathsf{vk}| \leq \mathsf{poly}(\lambda + \log k + s + \ell)$.

Fact 3 (BARGs for NP [CJJ21b, WW22, KLVW23, CGJ+23]). *Assuming any of (1) the plain LWE assumption, (2) the k-Lin assumption over pairing groups for any constant $k \in \mathbb{N}$, or (3) the (sub-exponential) DDH assumption in pairing-free groups, there exists a non-interactive batch argument for* NP.

Set Hiding with Extraction. For our main construction (Section 5), we require a slight strengthening of the somewhere extractability property from Definition 2. Our stronger set-hiding property essentially says that if the extraction key is programmed to extract either on $S_0 \subseteq [k]$ or $S_1 \subseteq [k]$, then the extracted witness on "common indices" $i^* \in S_0 \cap S_1$ is computationally indistinguishable in the two cases. This type of property is often referred to as a "no-signaling" extraction property [PR17, KPY19, GZ21, KVZ21, CJJ21b] (see the full version of this paper [NWW23]).

Index BARGs. An index BARG [CJJ21b] is a batch argument for the batch index language where the instance is *always* the tuple $(1, \ldots, k)$. Since the statements are the integers, they have a succinct description, so we can impose a stronger requirement on the running time of the Verify algorithm. We define this below:

Definition 3 (Index BARG [CJJ21b]). *An index BARG is a special case of a BARG where the instances $(x_1, \ldots, x_k)$ are restricted to the integers $(1, \ldots, k)$. In this setting, the* Gen *algorithm to the index BARG does* not *separately take in the instance length n as a separate input. Moreover, instead of providing $x_1, \ldots, x_k$ as input to the* Prove, Verify, *and* Extract *algorithms, we just give the single index k (in binary). Moreover, we require the* additional *succinctness property on the running time of* Verify*:*

- **Succinct verification time:** *There exists a fixed polynomial $\mathsf{poly}(\cdot)$ such that for all $\lambda, k, n, s, \ell \in \mathbb{N}$, all $(\mathsf{crs}, \mathsf{vk})$ in the support of $\mathsf{Gen}(1^\lambda, 1^k, 1^n, 1^s, 1^\ell)$ and all Boolean circuits $C\colon \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$ of size at most s, the running time of $\mathsf{Verify}(\mathsf{vk}, C, k, \cdot)$ is bounded by $\mathsf{poly}(\lambda + \log k + s + \ell)$.*

Monotone Policy BARG. Next, we recall the notion of a SNARG for monotone policy BatchNP [BBK+23], which we refer to more succinctly as a "monotone policy BARG." In this work, we just focus on the simplest notion of non-adaptive soundness.

Definition 4 (Monotone Policy BatchNP). *A Boolean circuit $P\colon \{0,1\}^k \to \{0,1\}$ is a monotone Boolean policy if P is a Boolean circuit comprised entirely of* AND *and* OR *gates. Let $C\colon \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$ be a Boolean circuit and $P\colon \{0,1\}^k \to \{0,1\}$ be a monotone Boolean policy. We define the monotone policy* BatchNP *language $\mathcal{L}_{\mathsf{MP\text{-}CSAT}}$ to be*

$$\mathcal{L}_{\mathsf{MP\text{-}CSAT}} = \left\{ (C, P, x_1, \ldots, x_k) \;\middle|\; \begin{array}{c} \exists w_1, \ldots, w_k \in \{0,1\}^h : \\ P\big(C(x_1, w_1), \ldots, C(x_k, w_k)\big) = 1 \end{array} \right\}.$$

Definition 5 (Monotone Policy BARG [BBK+23, adapted]). *A monotone policy BARG is a tuple* $\Pi_{\mathsf{MP\text{-}BARG}} = (\mathsf{Gen}, \mathsf{Prove}, \mathsf{Verify})$ *of efficient algorithms with the following syntax:*

- $\mathsf{Gen}(1^\lambda, 1^n, 1^{s_c}, 1^{s_p}) \to \mathsf{crs}$*: On input the security parameter* $\lambda \in \mathbb{N}$*, the instance size* $n \in \mathbb{N}$*, a bound on the size of the Boolean circuit* $s_c \in \mathbb{N}$*, and a bound on the size of the policy* $s_p \in \mathbb{N}$*, the generator algorithm outputs a common reference string* crs*.*
- $\mathsf{Prove}(\mathsf{crs}, C, P, (x_1, \ldots, x_k), (w_1, \ldots, w_k)) \to \pi$*: On input the common reference string* crs*, a Boolean circuit* $C\colon \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$*, a monotone Boolean policy* $P\colon \{0,1\}^k \to \{0,1\}$*, statements* $x_1, \ldots, x_k \in \{0,1\}^n$*, and witnesses* $w_1, \ldots, w_k \in \{0,1\}^h$*, the prove algorithm outputs a proof* π*.*
- $\mathsf{Verify}(\mathsf{crs}, C, P, (x_1, \ldots, x_k), \pi) \to b$*: On input the common reference string* crs*, a Boolean circuit* $C\colon \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$*, a monotone Boolean policy* $P\colon \{0,1\}^k \to \{0,1\}$*, statements* $x_1, \ldots, x_k \in \{0,1\}^n$*, and a proof* π*, the verification algorithm outputs a bit* $b \in \{0,1\}$*.*

Moreover, $\Pi_{\mathsf{MP\text{-}BARG}}$ *should satisfy the following properties:*

- ***Completeness:*** *For all* $\lambda, n, s_c, s_p \in \mathbb{N}$*, Boolean circuits* $C\colon \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$ *of size at most* s_c*, monotone Boolean policies* $P\colon \{0,1\}^k \to \{0,1\}$ *of size at most* s_p*, statements* $x = (x_1, \ldots, x_k) \in \{0,1\}^{kn}$ *and witnesses* $w = (w_1, \ldots, w_k) \in \{0,1\}^{kh}$ *where* $P(C(x_1, w_1), \ldots, C(x_k, w_k)) = 1$*, it holds that*

$$\Pr\left[\mathsf{Verify}(\mathsf{crs}, C, P, (x_1, \ldots, x_k), \pi) = 1 : \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda, 1^n, 1^{s_c}, 1^{s_p}) \\ \pi \leftarrow \mathsf{Prove}(\mathsf{crs}, C, P, x, w) \end{array}\right] = 1.$$

- ***Non-adaptive soundness:*** *For any adversary* $\mathcal{A}$*, define the non-adaptive soundness game as follows:*
  1. *On input the security parameter* 1^λ*, algorithm* $\mathcal{A}$ *starts by outputting the instance size* 1^n*, the bound on the size of the* NP *relation* 1^{s_c}*, the bound on the size of the policy* 1^{s_p}*, a Boolean circuit* $C\colon \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$ *of size at most* s_c*, a monotone Boolean circuit* $P\colon \{0,1\}^k \to \{0,1\}$ *of size at most* s_p*, and statements* $x_1, \ldots, x_k \in \{0,1\}^n$*.*
  2. *The challenger samples* $\mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda, 1^n, 1^{s_c}, 1^{s_p})$ *and gives it to* $\mathcal{A}$*.*
  3. *Algorithm* $\mathcal{A}$ *outputs a proof* π*.*
  4. *The output of the game is* $b = 1$ *if* $\mathsf{Verify}(\mathsf{crs}, C, P, (x_1, \ldots, x_k), \pi) = 1$ *and* $(C, P, (x_1, \ldots, x_k)) \notin \mathcal{L}_{\mathsf{MP\text{-}CSAT}}$*.*

  We say that $\Pi_{\mathsf{MP\text{-}BARG}}$ *is non-adaptively sound if for every efficient adversary* $\mathcal{A}$*, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that* $\Pr[b = 1] = \mathsf{negl}(\lambda)$ *in the non-adaptive soundness game.*
- **Succinctness:** *There exists a fixed polynomial* $\mathsf{poly}(\cdot)$ *such that for all* $\lambda, n, s_c, s_p \in \mathbb{N}$*, all* crs *in the support of* $\mathsf{Gen}(1^\lambda, 1^n, 1^{s_c}, 1^{s_p})$*, all Boolean circuits* $C\colon \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$ *of size at most* s_c*, and all monotone Boolean policies* $P\colon \{0,1\}^k \to \{0,1\}$ *of size* $|P| \le s_p$*, the following properties hold:*

- **Slightly succinct proofs:** *The proof* π *output by* $\mathsf{Prove}(\mathsf{crs}, C, P, \cdot, \cdot)$ *satisfies* $|\pi| \leq \mathsf{poly}(\lambda + s_c + \log s_p)$.
- **Succinct proofs:** *The proof* π *output by* $\mathsf{Prove}(\mathsf{crs}, C, P, \cdot, \cdot)$ *satisfies* $|\pi| \leq \mathsf{poly}(\lambda + s_c + \log |P|)$.

We provide additional discussion about monotone policy BARGs in the full version of this paper [NWW23].

3 Zero-Fixing Hash Functions

In this section, we formally introduce the notion of a zero-fixing hash function. As we show in Section 4, we can combine a zero-fixing hash function with a vanilla BARG to obtain a monotone policy BARG. Recall from Section 1.2 that a zero-fixing hash function is a keyed hash function that supports succinct local openings. Moreover, the hash key is associated with a set of indices $S \subseteq [n]$, where n is the input length. Moreover, there is a trapdoor td associated with the hash key hk that can be used to decide whether a hash digest dig is $\mathsf{Matching}$ or $\mathsf{NotMatching}$ on the set S. The zero-fixing security requirement then says that if the extractor outputs $\mathsf{Matching}$ for a digest dig, it must be computationally hard to open dig to a 1 on any index $i \in S$.

As discussed in Section 1.2, our zero-fixing hash function is similar to the predicate-extractable hash function for bit-fixing predicates from [BBK+23]. A key distinction is that when the extraction algorithm outputs $\mathsf{NotMatching}$, the predicate-extractable hash function also outputs an index $i \in [n]$ where it is computationally infeasible to open the digest to a 1. In contrast, with our zero-fixing hash function, the extraction algorithm only outputs a single $\mathsf{Matching}$ or $\mathsf{NotMatching}$ flag. At the same time, we require our zero-fixing hash functions to satisfy additional security requirements that were not required in [BBK+23]. These additional security properties are necessary for our construction of monotone policy BARGs (Section 4). We now give the formal definition:

Definition 6 (Zero-Fixing Hash Function). *A* zero-fixing hash function *is a tuple of polynomial-time algorithms* $\Pi_{\mathsf{H}} = (\mathsf{Setup}, \mathsf{Hash}, \mathsf{ProveOpen}, \mathsf{VerOpen}, \mathsf{Extract}, \mathsf{ValidateDigest})$ *with the following syntax:*

- $\mathsf{Setup}(1^\lambda, 1^n, S) \to (\mathsf{hk}, \mathsf{vk}, \mathsf{td})$*: On input a security parameter* λ*, an input length* n*, and a set* $S \subseteq [n]$*, the setup algorithm outputs a hash key* hk*, a verification key* vk *and a trapdoor* td*. We implicitly assume that* hk *includes* λ *and* n*.*
- $\mathsf{Hash}(\mathsf{hk}, x) \to \mathsf{dig}$*: On input a hash key* hk *and a string* $x \in \{0,1\}^n$*, the hash algorithm outputs a digest* dig*. This algorithm is deterministic.*
- $\mathsf{ValidateDigest}(\mathsf{vk}, \mathsf{dig}) \to b$*: On input a hash key* vk *and a digest* dig*, the digest validation algorithm outputs a bit* $b \in \{0,1\}$*. This algorithm is deterministic.*
- $\mathsf{ProveOpen}(\mathsf{hk}, x, i) \to \sigma$*: On input a hash key* hk*, a string* $x \in \{0,1\}^n$ *and an index* $i \in [n]$*, the prove algorithm outputs an opening* σ*.*

- $\mathsf{VerOpen}(\mathsf{vk}, \mathsf{dig}, i, b, \sigma) \to b'$*: On input a hash key* vk*, a digest* dig*, an index* $i \in [n]$*, a bit* $b \in \{0,1\}$ *and an opening* σ*, the verification algorithm outputs a bit* $b' \in \{0,1\}$*. The verification algorithm is deterministic.*
- $\mathsf{Extract}(\mathsf{td}, \mathsf{dig}) \to m$*: On input a trapdoor* td *and a digest* dig*, the extraction algorithm outputs a value* $m \in \{\mathsf{Matching}, \mathsf{NotMatching}\}$*. This algorithm is deterministic.*

We require Π_{H} *satisfy the following efficiency and correctness properties:*

- ***Succinctness:*** *There exists a universal polynomial* $\mathsf{poly}(\cdot)$ *such that for all parameters* $\lambda, n \in \mathbb{N}$*, all* $(\mathsf{hk}, \mathsf{vk}, \mathsf{td})$ *in the support of* $\mathsf{Setup}(1^\lambda, 1^n, \cdot)$*, all inputs* $x \in \{0,1\}^n$ *and all indices* $i \in [n]$*, the following properties hold:*
  - **Succinct verification key:** $|\mathsf{vk}| \leq \mathsf{poly}(\lambda + \log n)$.
  - **Succinct digest:** *The digest* dig *output by* $\mathsf{Hash}(\mathsf{hk}, x)$ *satisfies* $|\mathsf{dig}| \leq \mathsf{poly}(\lambda + \log n)$.
  - **Succinct openings:** *The opening* σ *output by* $\mathsf{ProveOpen}(\mathsf{hk}, x, i)$ *satisfies* $|\sigma| \leq \mathsf{poly}(\lambda + \log n)$.
  - **Succinct verification:** *The running time of* $\mathsf{VerOpen}(\mathsf{vk}, \cdot)$ *is* $\mathsf{poly}(\lambda + \log n)$.
- **Correctness:** *For all* $\lambda, n \in \mathbb{N}$*, every* $x \in \{0,1\}^n$*, and every* $i \in [n]$*, the following properties hold:*
  - ***Opening correctness:***

$$\Pr\left[\mathsf{VerOpen}(\mathsf{vk}, \mathsf{dig}, i, x_i, \sigma) = 1 : \begin{array}{c}(\mathsf{hk}, \mathsf{vk}, \mathsf{td}) \leftarrow \mathsf{Setup}(1^\lambda, 1^n, \varnothing) \\ \mathsf{dig} \leftarrow \mathsf{Hash}(\mathsf{hk}, x) \\ \sigma \leftarrow \mathsf{ProveOpen}(\mathsf{hk}, x, i)\end{array}\right] = 1.$$

  - ***Digest correctness:***

$$\Pr\left[\mathsf{ValidateDigest}(\mathsf{vk}, \mathsf{dig}) = 1 : \begin{array}{c}(\mathsf{hk}, \mathsf{vk}, \mathsf{td}) \leftarrow \mathsf{Setup}(1^\lambda, 1^n, \varnothing) \\ \mathsf{dig} \leftarrow \mathsf{Hash}(\mathsf{hk}, x)\end{array}\right] = 1.$$

We additionally require the following security properties:

- ***Set hiding:*** *For a bit* $b \in \{0,1\}$ *and an adversary* $\mathcal{A}$*, we define the set hiding game* $\mathsf{ExptSH}_{\mathcal{A}}(\lambda, b)$ *as follows:*
  1. *On input* 1^λ*, the adversary* $\mathcal{A}$ *outputs* 1^n *and a set* $S \subseteq [n]$*.*
  2. *If* $b = 0$*, the challenger samples* $(\mathsf{hk}, \mathsf{vk}, \mathsf{td}) \leftarrow \mathsf{Setup}(1^\lambda, 1^n, \varnothing)$ *and if* $b = 1$*, the challenger samples* $(\mathsf{hk}, \mathsf{vk}, \mathsf{td}) \leftarrow \mathsf{Setup}(1^\lambda, 1^n, S)$*. It gives* $(\mathsf{hk}, \mathsf{vk})$ *to* $\mathcal{A}$*.*
  3. *Algorithm* $\mathcal{A}$ *outputs a bit* b' *which is the output of the experiment.*

  The hash function satisfies set binding if for all efficient adversaries $\mathcal{A}$*, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that*

$$|\Pr[\mathsf{ExptSH}_{\mathcal{A}}(\lambda, 0) = 1] - \Pr[\mathsf{ExptSH}_{\mathcal{A}}(\lambda, 1) = 1]| = \mathsf{negl}(\lambda).$$

- ***One-Sided Index hiding with extracted guess:*** *For an adversary* $\mathcal{A}$ *and a bit* $b \in \{0,1\}$*, we define the index hiding with extracted guess game* $\mathsf{ExptIHE}_{\mathcal{A}}(\lambda, b)$ *as follows:*

1. *On input* 1^λ*, algorithm* $\mathcal{A}$ *outputs* 1^n*, a set* $S \subseteq [n]$*, and an index* $i^* \in S$*.*
2. *If* $b = 0$*, the challenger samples* $(\mathsf{hk}, \mathsf{vk}, \mathsf{td}) \leftarrow \mathsf{Setup}(1^\lambda, 1^n, S \setminus \{i^*\})$*. Otherwise, it samples* $(\mathsf{hk}, \mathsf{vk}, \mathsf{td}) \leftarrow \mathsf{Setup}(1^\lambda, 1^n, S)$*. The challenger sends* $(\mathsf{hk}, \mathsf{vk})$ *to* $\mathcal{A}$*.*
3. *Algorithm* $\mathcal{A}$ *outputs a digest* dig *and an opening* σ*.*
4. *The output of the experiment is* 1 *if* $\mathsf{VerOpen}(\mathsf{hk}, \mathsf{dig}, i^*, 0, \sigma) = 1$ *and* $\mathsf{Extract}(\mathsf{td}, \mathsf{dig})$ *outputs* Matching*. Otherwise, the output is* 0*.*

The hash function satisfies index hiding with extracted guess if for all efficient adversaries $\mathcal{A}$*, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that*

$$\Pr[\mathsf{ExptIHE}_{\mathcal{A}}(\lambda, 1) = 1] \geq \Pr[\mathsf{ExptIHE}_{\mathcal{A}}(\lambda, 0) = 1] - \mathsf{negl}(\lambda).$$

- ***Zero fixing:*** *For an adversary* $\mathcal{A}$*, we define the adaptive zero-fixing game* $\mathsf{ExptZF}_{\mathcal{A}}(\lambda)$ *as follows:*
  1. *On input* 1^λ*, algorithm* $\mathcal{A}$ *outputs* 1^n *and a set* $S \subseteq [n]$*.*
  2. *The challenger samples* $(\mathsf{hk}, \mathsf{vk}, \mathsf{td}) \leftarrow \mathsf{Setup}(1^\lambda, 1^n, S)$ *and gives* $(\mathsf{hk}, \mathsf{vk})$ *to* $\mathcal{A}$*.*
  3. *Algorithm* $\mathcal{A}$ *outputs a digest* dig*, an index* $i \in S$ *and an opening* σ*.*
  4. *The output of the experiment is* 1 *if* $\mathsf{VerOpen}(\mathsf{hk}, \mathsf{dig}, i, 1, \sigma) = 1$ *and* $\mathsf{Extract}(\mathsf{td}, \mathsf{dig})$ *outputs* Matching*. Otherwise, the output is* 0*.*

  The hash function satisfies zero-fixing if for all efficient adversaries $\mathcal{A}$*, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that* $\Pr[\mathsf{ExptZF}_{\mathcal{A}}(\lambda) = 1] = \mathsf{negl}(\lambda)$*.*
- ***Extractor validity:*** *For an adversary* $\mathcal{A}$*, we define the extractor validity game* $\mathsf{ExptEV}_{\mathcal{A}}(\lambda)$ *as follows:*
  1. *On input* 1^λ*, the adversary* $\mathcal{A}$ *outputs* 1^n*.*
  2. *The challenger samples* $(\mathsf{hk}, \mathsf{vk}, \mathsf{td}) \leftarrow \mathsf{Setup}(1^\lambda, 1^n, \varnothing)$ *and sends* $(\mathsf{hk}, \mathsf{vk})$ *to the adversary.*
  3. *Algorithm* $\mathcal{A}$ *outputs a digest* dig*.*
  4. *The output of the experiment is* 1 *if* $\mathsf{ValidateDigest}(\mathsf{hk}, \mathsf{dig}) = 1$ *and* $\mathsf{Extract}(\mathsf{td}, \mathsf{dig})$ *outputs* NotMatching*. Otherwise, the output is* 0*.*

  The hash function satisfies the extractor validity property if for every efficient adversary $\mathcal{A}$*, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that*

$$\Pr[\mathsf{ExptEV}_{\mathcal{A}}(\lambda) = 1] = \mathsf{negl}(\lambda).$$

4 Constructing Monotone Policy BARGs

In this section, we describe how to construct monotone policy BARGs from a standard batch argument for NP together with a zero-fixing hash function. We start by defining the conventions we use for describing Boolean circuits.

Definition 7 (Monotone Circuit Wire Indexing). *Let* $P\colon \{0,1\}^k \to \{0,1\}$ *be a monotone Boolean circuit consisting exclusively of* AND *and* OR *gates with fan-in two. Let* s *be the size of* P *(i.e., the number of wires in* P*). A topological indexing of the wires of* C *is an assignment of an index* $i \in [s]$ *to each wire in* P *with the following properties:*

- **Input wire:** *For $i \in [k]$, the i^{th} input to P is associated with the index i.*
- **Output wire:** *The output wire is associated with the index s.*
- **Intermediate wires:** *The intermediate wires are associated with an index $i \in \{k+1, \ldots, s-1\}$ with the property that the value of index i is completely determined by the values of the wires with indices $j_{i,1}, j_{i,2} \in \{1, \ldots, i-1\}$.*

Every monotone circuit P has a canonical topological indexing that can be computed efficiently (e.g., by applying a deterministic topological sort to the wires of P).

Definition 8 (Layered Monotone Circuit). *Let $P\colon \{0,1\}^k \to \{0,1\}$ be a (monotone) Boolean circuit of size s. We denote by $L_P(i)$ the* layer *of the wire i and define it as follows:*

- *If $i \in [k]$ (i.e., an input wire), then $L_P(i) = 1$.*
- *If $i > k$ then $L_P(i) = 1 + \max\{L_P(j_{i,1}), L_P(j_{i,2})\}$, where $j_{i,1}, j_{i,2}$ are the indices of the input wires to the gate that computes the value of wire i.*

The depth of the circuit is defined to be the layer associated with the output wire: $d = L_P(s)$. A circuit is layered *if for every $i \in \{k+1, \ldots, s\}$, it holds that $L_P(j_{i,1}) = L_P(j_{i,2})$. For a layer index $\ell \in [d]$, we define $\mathsf{layer}_\ell(P) = \{i \in [s] : L_P(i) = \ell\}$ to be the set of wire indices in layer ℓ of the circuit.*

Remark 1 (Layered Monotone Circuit). Every monotone circuit $P\colon \{0,1\}^k \to \{0,1\}$ of size s can be converted into a layered monotone circuit of size $\mathsf{poly}(s)$. Thus, without loss of generality, we exclusively consider layered monotone circuits in the remainder of this work.

4.1 Monotone Policy BARG Construction

We now describe our construction of a monotone policy BARG for NP.

Construction 4 (Monotone Policy BARG). Let $\Pi'_{\mathsf{BARG}} = (\mathsf{Gen}', \mathsf{Prove}', \mathsf{Verify}', \mathsf{TrapGen}', \mathsf{Extract}')$ be a somewhere extractable BARG for Boolean circuit satisfiability. Let $\Pi_{\mathsf{H}} = (\mathsf{H.Setup}, \mathsf{H.Hash}, \mathsf{H.ProveOpen}, \mathsf{H.VerOpen}, \mathsf{H.Extract}, \mathsf{H.ValidateDigest})$ be a zero-fixing hash function. We construct a monotone policy BARG $\Pi_{\mathsf{MP\text{-}BARG}} = (\mathsf{Gen}, \mathsf{Prove}, \mathsf{Verify})$ as follows:

- $\mathsf{Gen}(1^\lambda, 1^n, 1^{s_c}, 1^{s_p})$: On input the security parameter λ, the input length n, the bound on the size of the Boolean circuit s_c, and the bound on the size of the monotone policy s_p, the setup algorithm proceeds as follows:
  - Sample two hash keys

$$(\mathsf{hk}_0, \mathsf{vk}_0, \mathsf{td}_0) \leftarrow \mathsf{H.Setup}(1^\lambda, 1^{s_p}, \varnothing)$$
$$(\mathsf{hk}_1, \mathsf{vk}_1, \mathsf{td}_1) \leftarrow \mathsf{H.Setup}(1^\lambda, 1^{s_p}, \varnothing).$$

- Let s' be a bound on the size of the circuit that computes the relation $\mathcal{R}[C, k, s_p, \mathsf{vk}_0, \mathsf{vk}_1, \mathsf{dig}_0, \mathsf{dig}_1]$ from Fig. 1 when instantiated with an arbitrary Boolean circuit C of size at most s_c, an input length $k \leq s_p$ and digests $\mathsf{dig}_0, \mathsf{dig}_1$ associated with the hash and verification keys $(\mathsf{hk}_0, \mathsf{vk}_0)$ and $(\mathsf{hk}_1, \mathsf{vk}_1)$. Let $n' = 3 \cdot \lceil \log s_p \rceil + 1$ be the bound on the statement length. Sample $(\mathsf{crs}_{\mathsf{BARG}}, \mathsf{vk}_{\mathsf{BARG}}) \leftarrow \mathsf{Gen}'(1^\lambda, 1^{s_p}, 1^{n'}, 1^{s'})$.[7]

  It outputs $\mathsf{crs} = (\mathsf{crs}_{\mathsf{BARG}}, \mathsf{vk}_{\mathsf{BARG}}, \mathsf{hk}_0, \mathsf{hk}_1, \mathsf{vk}_0, \mathsf{vk}_1)$.

– $\mathsf{Prove}(\mathsf{crs}, C, P, (x_1, \ldots, x_k), (w_1, \ldots, w_k))$: On input $\mathsf{crs} = (\mathsf{crs}_{\mathsf{BARG}}, \mathsf{vk}_{\mathsf{BARG}}, \mathsf{hk}_0, \mathsf{hk}_1, \mathsf{vk}_0, \mathsf{vk}_1)$, a circuit $C\colon \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$, a monotone layered Boolean policy circuit $P\colon \{0,1\}^k \to \{0,1\}$, statements $x_1, \ldots, x_k \in \{0,1\}^n$, and witnesses $w_1, \ldots, w_k \in \{0,1\}^h$, the prove algorithm does the following:

  - Let s be the size of P. Index the wires of P under a canonical topological ordering (Definition 7). For each wire $i \in \{k+1, \ldots, s\}$, let $g_i \in \{\text{AND}, \text{OR}\}$ be its type. Let $j_{i,1}, j_{i,2} \in \{1, \ldots, i-1\}$ be the indices of the input wires to the gate i.
  - For each $i \in [s]$, let $\beta_i \in \{0,1\}$ be the value of wire i in the evaluation of P on input $(C(x_1, w_1), \ldots, C(x_k, w_k))$. For $i \in \{s+1, \ldots, s_p\}$, let $\beta_i = 0$. (This corresponds to "padding" the $s_p - s$ unused slots).
  - Compute the digest $\mathsf{dig}_0 \leftarrow \mathsf{H.Hash}(\mathsf{hk}_0, (\beta_1, \ldots, \beta_{s_p}))$ and the digest $\mathsf{dig}_1 \leftarrow \mathsf{H.Hash}(\mathsf{hk}_1, (\beta_1, \ldots, \beta_{s_p}))$.
  - For each $i \in [s]$ and each $b \in \{0,1\}$, compute

$$\sigma_i^{(b)} \leftarrow \mathsf{H.ProveOpen}(\mathsf{hk}_b, (\beta_1, \ldots, \beta_{s_p}), i).$$

  - Let C_{aug} be the circuit that computes $\mathcal{R}[C, k, s, \mathsf{vk}_0, \mathsf{vk}_1, \mathsf{dig}_0, \mathsf{dig}_1]$ shown in Fig. 1.
  - For each $i \in [s_p]$, construct the statement $\hat{x}_i$ and witness $\hat{w}_i$ as follows:
    * If $i \in [k]$, let $\hat{x}_i = (i, x_i)$ and $\hat{w}_i = \big(\beta_i, \sigma_i^{(0)}, \sigma_i^{(1)}, w_i\big)$.
    * If $i \in [k+1, s]$, let $\hat{x}_i = (i, (g_i, j_{i,1}, j_{i,2}))$ and

$$\hat{w}_i = \big(\beta_i, \sigma_i^{(0)}, \sigma_i^{(1)}, \big(\beta_{j_{i,1}}, \sigma_{j_{i,1}}^{(0)}, \sigma_{j_{i,1}}^{(1)}, \beta_{j_{i,2}}, \sigma_{j_{i,2}}^{(0)}, \sigma_{j_{i,2}}^{(1)}\big)\big).$$

    * If $i > s$, let $\hat{x}_i = \bot$ and $\hat{w}_i = \bot$.

    Essentially, there is an instance $\hat{x}_i$ associated with each wire i of P.
  - Compute the BARG proof

$$\pi_{\mathsf{BARG}} \leftarrow \mathsf{Prove}'(\mathsf{crs}_{\mathsf{BARG}}, C_{\mathsf{aug}}, (\hat{x}_1, \ldots, \hat{x}_{s_p}), (\hat{w}_1, \ldots, \hat{w}_{s_p}))$$

    and output $\pi = (\mathsf{dig}_0, \mathsf{dig}_1, \pi_{\mathsf{BARG}})$.

– $\mathsf{Verify}(\mathsf{crs}, C, P, (x_1, \ldots, x_k), \pi)$: On input a common reference string $\mathsf{crs} = (\mathsf{crs}_{\mathsf{BARG}}, \mathsf{vk}_{\mathsf{BARG}}, \mathsf{hk}_0, \mathsf{hk}_1, \mathsf{vk}_0, \mathsf{vk}_1)$, a Boolean circuit $C\colon \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$, a layered monotone Boolean policy $P\colon \{0,1\}^k \to \{0,1\}$, statements $x_1, \ldots, x_k \in \{0,1\}^n$, and a proof $\pi = (\mathsf{dig}_0, \mathsf{dig}_1, \pi_{\mathsf{BARG}})$, the verification algorithm does the following:

[7] Recall that when the bound on the extraction set parameter ℓ is not given, it defaults to the value 1.

- If $\mathsf{H.ValidateDigest}(\mathsf{vk}_0, \mathsf{dig}_0) = 0$ or $\mathsf{H.ValidateDigest}(\mathsf{vk}_1, \mathsf{dig}_1) = 0$, then output 0.
- Let s be the size of P. Index the wires of P under a canonical topological ordering (Definition 7). For each wire $i \in \{k+1, \ldots, s\}$, let $j_{i,1}, j_{i,2} \in \{1, \ldots, i-1\}$ be the indices of the input wires of the gate $g_i \in \{\text{AND}, \text{OR}\}$ that computes wire i. For each $i \in [s_p]$, construct the statement $\hat{x}_i$ as follows:
  - If $i \in [k]$, let $\hat{x}_i = (i, x_i)$.
  - If $i \in \{k+1, \ldots, s\}$, let $\hat{x}_i = (i, (g_i, j_{i,1}, j_{i,2}))$.
  - If $i > s$, let $\hat{x}_i = \perp$.
- Let C_{aug} be the circuit that computes $\mathcal{R}[C, k, s, \mathsf{vk}_0, \mathsf{vk}_1, \mathsf{dig}_0, \mathsf{dig}_1]$ from Fig. 1.
- Output $\mathsf{Verify}'(\mathsf{vk}_{\mathsf{BARG}}, C_{\mathsf{aug}}, (\hat{x}_1, \ldots, \hat{x}_{s_p}), \pi_{\mathsf{BARG}})$.

We defer the formal correctness and security proofs to the full version of this paper [NWW23].

Statement: index i and auxiliary statement x
Witness: value b, openings $(\sigma^{(0)}, \sigma^{(1)})$ and auxiliary witness w
Hard-Coded: circuit $C\colon \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$, the number of inputs k, the policy size s, the hash keys $\mathsf{hk}_0, \mathsf{hk}_1$, and the digests $\mathsf{dig}_0, \mathsf{dig}_1$

On input a statement (i, x) and a witness $(b, \sigma^{(0)}, \sigma^{(1)}, w)$:

- If $i \le k$, output 1 if all of the following conditions are met, otherwise output 0:
  - **Opening validity:** For all $\alpha \in \{0,1\}$, $\mathsf{H.VerOpen}\left(\mathsf{vk}_\alpha, \mathsf{dig}_\alpha, i, b, \sigma^{(\alpha)}\right) = 1$.
  - **Wire consistency:** $C(x, w) = b$.
- If $i \in \{k+1, \ldots, s\}$, parse $x = (g, j_1, j_2)$ where $g \in \{\text{AND}, \text{OR}\}$ and $j_1, j_2 \in \{1, \ldots, i-1\}$. Parse $w = (b_1, \sigma_1^{(0)}, \sigma_1^{(1)}, b_2, \sigma_2^{(0)}, \sigma_2^{(1)})$. Check each of the following conditions for $\alpha \in \{0,1\}$: Output 1 if all of the following conditions are met, otherwise output 0:
  - **Opening validity:** For all $\alpha \in \{0,1\}$, all of the following holds:
    - $\mathsf{H.VerOpen}\left(\mathsf{vk}_\alpha, \mathsf{dig}_\alpha, j_1, b_1, \sigma_1^{(\alpha)}\right) = 1$;
    - $\mathsf{H.VerOpen}\left(\mathsf{vk}_\alpha, \mathsf{dig}_\alpha, j_2, b_2, \sigma_2^{(\alpha)}\right) = 1$;
    - $\mathsf{H.VerOpen}\left(\mathsf{vk}_\alpha, \mathsf{dig}_\alpha, i, b, \sigma^{(\alpha)}\right) = 1$.
  - **Wire consistency:** $b = g(b_1, b_2)$.
  - **Output gate:** If $i = s$, check that $b = 1$.
- If $i > s$, then output 1.

Fig. 1. The relation $\mathcal{R}[C, k, s, \mathsf{vk}_0, \mathsf{vk}_1, \mathsf{dig}_0, \mathsf{dig}_1]$.

5 Generic Construction of Zero-Fixing Hash Functions

In this section, we show how to construct a zero-fixing hash function by combining an index BARG (Definition 3), an additively homomorphic encryption scheme with bounded support , and a vector encryption scheme with succinct local openings .

Binary Tree Indexing. In the following construction, we will work with complete binary trees. We will use the following procedure to associate a unique index with each node in the binary tree:

Definition 9 (Binary Tree Indexing). *Let $\mathcal{T}$ be a complete binary tree with $n = 2^k$ leaves. Then $\mathcal{T}$ contains exactly $2n - 1$ nodes. We associate a unique index $i \in [2n - 1]$ via the following procedure:*

- *First, associate the value $v = 1$ to the root node.*
- *If v is the value associated with a node, then associate values $2v$ and $2v + 1$ with its left and right child. Recursively apply this process to assign a value to every node in the tree.*
- *The index i associated with a node is defined to be $2n - v$, where v is the value associated with the node.*

By design, Definition 9 has the following properties:

- The leaf nodes are indexed 1 through n and the root node is indexed $2n - 1$.
- The index of every non-leaf node is greater than the index of its children.
- Given the index of any non-leaf node, we can efficiently compute the indices of its left and right child.

Construction 5 (Zero-Fixing Hash Function). Our construction will rely on the following building blocks:

- Let $\Pi'_{\mathsf{BARG}} = (\mathsf{Gen}', \mathsf{Prove}', \mathsf{Verify}', \mathsf{TrapGen}', \mathsf{Extract}')$ be a somewhere extractable *index* BARG (Definition 3).
- Let $\Pi_{\mathsf{HE}} = (\mathsf{HE.Gen}, \mathsf{HE.Enc}, \mathsf{HE.Dec}, \mathsf{HE.Add})$ be an additively homomorphic encryption scheme with bounded support . Let $\ell_{\mathsf{ct}}(\lambda, n)$ be a bound on the length of the ciphertexts output by either $\mathsf{HE.Enc}(\mathsf{pk}, \cdot)$ or $\mathsf{HE.Add}(\mathsf{pk}, \cdot, \cdot)$ for any $(\mathsf{sk}, \mathsf{pk})$ in the support of $\mathsf{HE.Gen}(1^\lambda, 1^n)$.
- Let $\Pi_{\mathsf{Com}} = (\mathsf{Com.Setup}, \mathsf{Com.Commit}, \mathsf{Com.Verify})$ be a vector commitment scheme with succinct local openings .

We construct a zero-fixing hash $\Pi_{\mathsf{H}} = (\mathsf{Setup}, \mathsf{Hash}, \mathsf{ProveOpen}, \mathsf{VerOpen}, \mathsf{Extract}, \mathsf{ValidateDigest})$. In the following description, we assume without loss of generality that the bound on the input length $n \in \mathbb{N}$ is a power of two (i.e., $n = 2^k$ for some integer $k \in \mathbb{N}$). Next, we define the following NP relation which we will be using in our construction:

We describe our construction below:

Statement: index $i \in [n]$
Witness: ciphertexts $\hat{v}^{(0)}, \hat{v}^{(1)}$, openings $\sigma^{(0)}, \sigma^{(1)}$, and an auxiliary witness $\tilde{w}$
Hard-coded: the common reference string $\mathsf{crs}_{\mathsf{Com}}$ for Π_{Com}, an index $i^* \in [n] \cup \{\perp\}$, a value $y \in \{0, 1, \perp\}$, and for each $b \in \{0, 1\}$, a public key pk_b for Π_{HE}, commitments $\mathsf{com}_{\mathsf{hk}}^{(b)}$ and two ciphertexts $\mathsf{ct}_{\mathsf{zero}}^{(b)}$, $\mathsf{ct}_{\mathsf{root}}^{(b)}$

On input a statement $i \in [n]$ and a witness $\left(\hat{v}^{(0)}, \hat{v}^{(1)}, \sigma^{(0)}, \sigma^{(1)}, \tilde{w}\right)$:

- **Leaf nodes:** If $i \in [n]$, then parse $\tilde{w} = \left(\hat{\mathsf{ct}}^{(0)}, \hat{\mathsf{ct}}^{(1)}, \sigma_{\mathsf{hk}}^{(0)}, \sigma_{\mathsf{hk}}^{(1)}\right)$. Output 1 if the following conditions hold:
  1. **Opening to ciphertext:** for $b \in \{0, 1\}$, $\mathsf{Com.Verify}\left(\mathsf{crs}_{\mathsf{Com}}, \mathsf{com}_b, i, \hat{v}^{(b)}, \sigma^{(b)}\right) = 1$.
  2. **Opening to ciphertext in hk:** for $b \in \{0, 1\}$, $\mathsf{Com.Verify}\left(\mathsf{crs}_{\mathsf{Com}}, \mathsf{com}_{\mathsf{hk}}^{(b)}, i, \hat{\mathsf{ct}}^{(b)}, \sigma_{\mathsf{hk}}^{(b)}\right) = 1$.
  3. **Consistent choice of ciphertexts:** $\left(\hat{v}^{(0)} = \mathsf{ct}_{\mathsf{zero}}^{(0)} \wedge \hat{v}^{(1)} = \mathsf{ct}_{\mathsf{zero}}^{(1)}\right)$ or $\left(\hat{v}^{(0)} = \hat{\mathsf{ct}}^{(0)} \wedge \hat{v}^{(1)} = \hat{\mathsf{ct}}^{(1)}\right)$.
  4. **Validity of ciphertext at target index:** If $i = i^*$, then *additionally* check that:
  $$\hat{v}^{(b)} = \begin{cases} \mathsf{ct}_{\mathsf{zero}}^{(b)} & y = 0 \\ \hat{\mathsf{ct}}^{(b)} & y = 1. \end{cases}$$
  If any of these conditions are not satisfied, output 0.
- **Non-leaf nodes:** If $i \in [n+1, 2n-1]$, then parse $\tilde{w} = (\tilde{w}_{\mathrm{L}}, \tilde{w}_{\mathrm{R}})$, where $\tilde{w}_d = \left(\hat{v}_d^{(0)}, \hat{v}_d^{(1)}, \sigma_d^{(0)}, \sigma_d^{(1)}\right)$ for $d \in \{\mathrm{L}, \mathrm{R}\}$. Output 1 if all of the following conditions hold for all $b \in \{0, 1\}$:
  1. **Opening to ciphertext:** $\mathsf{Com.Verify}\left(\mathsf{crs}_{\mathsf{Com}}, \mathsf{com}_b, i, \hat{v}^{(b)}, \sigma^{(b)}\right) = 1$.
  2. **Opening to child ciphertexts:** $\mathsf{Com.Verify}(\mathsf{crs}_{\mathsf{Com}}, \mathsf{com}_b, i_{\mathrm{L}}, \hat{v}_{\mathrm{L}}^{(b)}, \sigma_{\mathrm{L}}^{(b)}) = 1$ and $\mathsf{Com.Verify}(\mathsf{crs}_{\mathsf{Com}}, \mathsf{com}_b, i_{\mathrm{R}}, \hat{v}_{\mathrm{R}}^{(b)}, \sigma_{\mathrm{R}}^{(b)}) = 1$, where i_{L} and i_{R} are the indices of the left and right child of i (according to the indexing scheme from Definition 9).
  3. **Correctness of evaluation:** $\hat{v}^{(b)} = \mathsf{HE.Add}\left(\mathsf{pk}_b, \hat{v}_{\mathrm{L}}^{(b)}, \hat{v}_{\mathrm{R}}^{(b)}\right)$.
  4. **Validity of root:** If $i = 2n - 1$ then $\hat{v}^{(b)} = \mathsf{ct}_{\mathsf{root}}^{(b)}$.
  If any of these conditions are not satisfied, output 0.

Fig. 2. The index relation $\mathcal{R}\left[\mathsf{crs}_{\mathsf{Com}}, \left\{\mathsf{pk}_b, \mathsf{com}_{\mathsf{hk}}^{(b)}, \mathsf{com}_b, \mathsf{ct}_{\mathsf{zero}}^{(b)}, \mathsf{ct}_{\mathsf{root}}^{(b)}\right\}_{b \in \{0,1\}}, i^*, y\right]$.

- $\mathsf{Setup}(1^\lambda, 1^n, S)$: On input a security parameter λ, the input length $n = 2^k$, and a set $S \subseteq [n]$, the setup algorithm starts by sampling the following:
  - Sample two key pairs: $(\mathsf{sk}_0, \mathsf{pk}_0) \leftarrow \mathsf{HE.Gen}(1^\lambda, 1^n)$ and $(\mathsf{sk}_1, \mathsf{pk}_1) \leftarrow \mathsf{HE.Gen}(1^\lambda, 1^n)$.
  - Sample the CRS for the commitment scheme with block length $\ell_{\mathsf{ct}}(\lambda, n)$ and up to $2n - 1$ blocks: $\mathsf{crs}_{\mathsf{Com}} \leftarrow \mathsf{Com.Setup}(1^\lambda, 1^{\ell_{\mathsf{ct}}(\lambda,n)}, 2n-1)$.
  - Sample the CRS for a BARG: $(\mathsf{crs}_{\mathsf{BARG}}, \mathsf{vk}_{\mathsf{BARG}}) \leftarrow \mathsf{Gen}'(1^\lambda, 1^{2n-1}, 1^s, 1^3)$, where s is a bound on the size of the circuit computing the index relation from Fig. 2. Here, the CRS is extractable on up to 3 positions. Note

that since Π_{BARG} is an *index* BARG, Gen' does *not* separately take the statement length as input (Definition 3).

Next, for each $b \in \{0,1\}$, construct an encryption of 0: $\mathsf{ct}_{\mathsf{zero}}^{(b)} \leftarrow \mathsf{HE.Enc}(\mathsf{pk}_b, 0)$. Next, for each $i \in S$ and $b \in \{0,1\}$, construct the hash key ciphertexts as follows:

- If $i \in S$, compute $\mathsf{ct}_i^{(b)} \leftarrow \mathsf{HE.Enc}(\mathsf{pk}_b, 1)$.
- If $i \notin S$, compute $\mathsf{ct}_i^{(b)} \leftarrow \mathsf{HE.Enc}(\mathsf{pk}_b, 0)$.

Next, the setup algorithm constructs a commitment to the ciphertexts associated with the hash key. Specifically, for each $b \in \{0,1\}$, it computes

$$\left(\mathsf{com}_{\mathsf{hk}}^{(b)}, \sigma_{\mathsf{hk},1}^{(b)}, \ldots, \sigma_{\mathsf{hk},n}^{(b)}\right) \leftarrow \mathsf{Com.Commit}\left(\mathsf{crs}_{\mathsf{Com}}, \left(\mathsf{ct}_1^{(b)}, \ldots, \mathsf{ct}_n^{(b)}\right)\right).$$

Finally, the setup algorithm constructs the hash key hk, the verification key vk, and the trapdoor td as follows:

$$\mathsf{hk} = \left(\mathsf{crs}_{\mathsf{Com}}, \mathsf{crs}_{\mathsf{BARG}}, \left\{\mathsf{pk}_b, \mathsf{ct}_{\mathsf{zero}}^{(b)}, \mathsf{ct}_1^{(b)}, \ldots, \mathsf{ct}_n^{(b)}, \sigma_{\mathsf{hk},1}^{(b)}, \ldots, \sigma_{\mathsf{hk},n}^{(b)}\right\}_{b\in\{0,1\}}\right) \tag{5.1}$$

$$\mathsf{vk} = \left(\mathsf{crs}_{\mathsf{Com}}, \mathsf{vk}_{\mathsf{BARG}}, \mathsf{pk}_0, \mathsf{pk}_1, \mathsf{ct}_{\mathsf{zero}}^{(0)}, \mathsf{ct}_{\mathsf{zero}}^{(1)}, \mathsf{com}_{\mathsf{hk}}^{(0)}, \mathsf{com}_{\mathsf{hk}}^{(1)}\right) \tag{5.2}$$

$$\mathsf{td} = (\mathsf{sk}_0, \mathsf{sk}_1). \tag{5.3}$$

– $\mathsf{Hash}(\mathsf{hk}, x)$: On input a hash key hk (parsed as in Eq. (5.1)) and a string $x \in \{0,1\}^n$, the hashing algorithm proceeds as follows:

- Construct two complete binary trees $\mathcal{T}_0, \mathcal{T}_1$, each with n leaves. For each tree $\mathcal{T}_b$, we assign a ciphertext $v_i^{(b)}$ to each node $i \in [2s-1]$ in the tree as follows (where the nodes are indexed using Definition 9):
  - If $i \in [n]$, let $v_i^{(b)} \leftarrow \mathsf{ct}_{\mathsf{zero}}^{(b)}$ if $x_i = 0$ and $v_i^{(b)} \leftarrow \mathsf{ct}_i^{(b)}$ if $x_i = 1$.
  - For each internal node $i \in [n+1, 2n-1]$, compute the ciphertext $v_i^{(b)} \leftarrow \mathsf{HE.Add}\left(\mathsf{pk}_b, v_{i_{\mathrm{L}}}^{(b)}, v_{i_{\mathrm{R}}}^{(b)}\right)$, where i_{L} and i_{R} are the indices associated with the left and right child of node i under the canonical tree indexing scheme (Definition 9).
- For $b \in \{0,1\}$, construct commitments

$$(\mathsf{com}_b, \sigma_1^{(b)}, \ldots, \sigma_{2n-1}^{(b)}) \leftarrow \mathsf{Com.Commit}(\mathsf{crs}_{\mathsf{Com}}, (v_1^{(b)}, \ldots, v_{2n-1}^{(b)}))$$

  to the ciphertexts associated with $\mathcal{T}_b$.
- For $b \in \{0,1\}$, let $\mathsf{ct}_{\mathsf{root}}^{(b)} = v_{2n-1}^{(b)}$ (i.e., the ciphertext associated with the root of $\mathcal{T}_b$). Let $C_\perp$ be the circuit that computes the following instantiation of the relation from Fig. 2:

$$\mathcal{R}\left[\mathsf{crs}_{\mathsf{Com}}, \left\{\mathsf{pk}_b, \mathsf{com}_{\mathsf{hk}}^{(b)}, \mathsf{com}_b, \mathsf{ct}_{\mathsf{zero}}^{(b)}, \mathsf{ct}_{\mathsf{root}}^{(b)}\right\}_{b\in\{0,1\}}, \perp, \perp\right].$$

- For each $i \in [2n-1]$, let $\tau_i = \left(v_i^{(0)}, v_i^{(1)}, \sigma_i^{(0)}, \sigma_i^{(1)}\right)$ be the opening for the ciphertexts associated with node i in $\mathcal{T}_0$ and $\mathcal{T}_1$. Then, for each $i \in [2s-1]$, define the auxiliary witness $\tilde{w}_i$ to be

* If $i \in [n]$ then $\tilde{w}_i = \left(\mathsf{ct}_i^{(0)}, \mathsf{ct}_i^{(1)}, \sigma_{\mathsf{hk},i}^{(0)}, \sigma_{\mathsf{hk},i}^{(1)}\right)$.
* If $i \in [n+1, 2n-1]$ then $\tilde{w}_i = (\tau_{i_\mathrm{L}}, \tau_{i_\mathrm{R}})$ where $i_\mathrm{L}, i_\mathrm{R}$ are the indices of the left and right child of node i, respectively.

Finally, let $w_i = (\tau_i, \tilde{w}_i)$ for each $i \in [2n-1]$. Compute the BARG proof $\pi_{\mathsf{dig}} \leftarrow \mathsf{Prove}'(\mathsf{crs}_{\mathsf{BARG}}, C_\perp, 2n-1, (w_1, \ldots, w_{2n-1}))$.

- Output the digest

$$\mathsf{dig} = \left(\mathsf{ct}_{\mathsf{root}}^{(0)}, \mathsf{ct}_{\mathsf{root}}^{(1)}, \mathsf{com}_0, \mathsf{com}_1, \pi_{\mathsf{dig}}\right).$$

– $\mathsf{ProveOpen}(\mathsf{hk}, x, i^*)$: On input a hash key hk (parsed as in Eq. (5.1)), a string $x \in \{0,1\}^n$ and an index $i^* \in [n]$, the opening algorithm proceeds as follows:
- Let $C_{i^*, x_{i^*}}$ be the circuit that computes the following instantiation of the relation from Fig. 2:

$$\mathcal{R}\left[\mathsf{crs}_{\mathsf{Com}}, \left\{\mathsf{pk}_b, \mathsf{com}_{\mathsf{hk}}^{(b)}, \mathsf{com}_b, \mathsf{ct}_{\mathsf{zero}}^{(b)}, \mathsf{ct}_{\mathsf{root}}^{(b)}\right\}_{b\in\{0,1\}}, i^*, x_{i^*}\right].$$

- Compute the witnesses w_i for each $i \in [2n-1]$ using the same procedure as in the Hash algorithm.
- Output the opening $\sigma \leftarrow \mathsf{Prove}'(\mathsf{crs}_{\mathsf{BARG}}, C_{i^*, x_{i^*}}, 2n-1, (w_1, \ldots, w_{2n-1}))$

– $\mathsf{VerOpen}(\mathsf{vk}, \mathsf{dig}, i, b, \sigma)$: On input the verification key vk (parsed according to Eq. (5.2)), a digest $\mathsf{dig} = \left(\mathsf{ct}_{\mathsf{root}}^{(0)}, \mathsf{ct}_{\mathsf{root}}^{(1)}, \mathsf{com}_0, \mathsf{com}_1, \pi_{\mathsf{dig}}\right)$, an index $i^* \in [n]$, a bit $b \in \{0,1\}$ and an opening σ, the verification algorithm outputs $\mathsf{Verify}'(\mathsf{vk}_{\mathsf{BARG}}, C_{i^*,b}, 2n-1, \sigma)$ where $C_{i^*,b}$ is the circuit computing the following relation from Fig. 2:

$$\mathcal{R}\left[\mathsf{crs}_{\mathsf{Com}}, \left\{\mathsf{pk}_b, \mathsf{com}_{\mathsf{hk}}^{(b)}, \mathsf{com}_b, \mathsf{ct}_{\mathsf{zero}}^{(b)}, \mathsf{ct}_{\mathsf{root}}^{(b)}\right\}_{b\in\{0,1\}}, i^*, b\right].$$

– $\mathsf{Extract}(\mathsf{td}, \mathsf{dig})$: On input a trapdoor $\mathsf{td} = (\mathsf{sk}_0, \mathsf{sk}_1)$ and a digest $\mathsf{dig} = \left(\mathsf{ct}_{\mathsf{root}}^{(0)}, \mathsf{ct}_{\mathsf{root}}^{(1)}, \mathsf{com}_0, \mathsf{com}_1, \pi_{\mathsf{dig}}\right)$, the extraction algorithm outputs $\mathsf{Matching}$ if $\mathsf{HE.Dec}(\mathsf{sk}_0, \mathsf{ct}_{\mathsf{root}}^{(0)}) = 0$. Otherwise, the algorithm outputs $\mathsf{NotMatching}$.

– $\mathsf{ValidateDigest}(\mathsf{vk}, \mathsf{dig})$: On input the verification key vk (parsed according to Eq. (5.2)) and a digest $\mathsf{dig} = \left(\mathsf{ct}_{\mathsf{root}}^{(0)}, \mathsf{ct}_{\mathsf{root}}^{(1)}, \mathsf{com}_0, \mathsf{com}_1, \pi_{\mathsf{dig}}\right)$, the digest-validation algorithm outputs $\mathsf{Verify}'(\mathsf{vk}_{\mathsf{BARG}}, C_\perp, 2n-1, \pi_{\mathsf{dig}})$ where $C_\perp$ is the circuit computing the following relation from Fig. 2:

$$\mathcal{R}\left[\mathsf{crs}_{\mathsf{Com}}, \left\{\mathsf{pk}_b, \mathsf{com}_{\mathsf{hk}}^{(b)}, \mathsf{com}_b, \mathsf{ct}_{\mathsf{zero}}^{(b)}, \mathsf{ct}_{\mathsf{root}}^{(b)}\right\}_{b\in\{0,1\}}, \perp, \perp\right].$$

Correctness and Security Analysis. Due to space limitations, we defer the correctness and security analysis of Construction 5 to the full version of this paper [NWW23].

Corollary 1 (Zero-Fixing Hash Functions). *Assuming any of (1) the plain LWE assumption, (2) the k-Lin assumption over pairing groups for any constant k, or (3) the (sub-exponential) DDH assumption in pairing-free groups, there exists a zero-fixing hash function.*

Theorem 1 now follows in conjunction with our generic construction (Construction 4). Due to space limitations, we defer the security analysis of Construction 4 to the full version of this paper [NWW23].

Acknowledgments. We thank Yuval Ishai for helpful pointers on batch arguments. Brent Waters is supported by NSF CNS-1908611, CNS-2318701, and a Simons Investigator award. David J. Wu is supported by NSF CNS-2151131, CNS-2140975, CNS-2318701, a Microsoft Research Faculty Fellowship, and a Google Research Scholar award.

References

BBK+23. Brakerski, Z., Brodsky, M.F., Kalai, Y.T., Lombardi, A., Paneth, O.: SNARGs for monotone policy batch NP. In: CRYPTO, pp. 252–283 (2023)

BCJP24. Brodsky, M.F., Choudhuri, A.R., Jain, A., Paneth, O.: Monotone-policy aggregate signatures. In: EUROCRYPT (2024)

BKP22. Ben-David, S., Kalai, Y.T., Paneth, O.: Verifiable private information retrieval. In: TCC, pp. 3–32 (2022)

CF13. Catalano, D., Fiore, D.: Vector commitments and their applications. In: PKC, pp. 55–72 (2013)

CGJ+23. Choudhuri, A.R., Garg, S., Jain, A., Jin, Z., Zhang, J.: Correlation intractability and SNARGs from sub-exponential DDH. In: CRYPTO, pp. 635–668 (2023)

CJJ21a. Choudhuri, A.R., Jain, A., Jin, Z.: Non-interactive batch arguments for NP from standard assumptions. In: CRYPTO, pp. 394–423 (2021)

CJJ21b. Choudhuri, A.R., Jain, A., Jin, Z.: SNARGs for P from LWE. In: FOCS, pp. 68–79 (2021)

DGKV22. Devadas, L., Goyal, R., Kalai, Y., Vaikuntanathan, V.: Rate-1 non-interactive arguments for batch-NP and applications. In: FOCS, pp. 1057–1068 (2022)

FWW23. Freitag, C., Waters, B., Wu, D.J.: How to use (plain) witness encryption: registered ABE, flexible broadcast, and more. In: CRYPTO, pp. 498–531 (2023)

Gam84. El Gamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. In: CRYPTO, pp. 10–18 (1984)

GVW19. Goyal, R., Vusirikala, S., Waters, B.: Collusion resistant broadcast and trace from positional witness encryption. In: PKC, pp. 3–33 (2019)

GZ21. González, A., Zacharakis, A.: Succinct publicly verifiable computation. IACR Cryptol. ePrint Arch., p. 353 (2021)

HJKS22. Hulett, J., Jawale, R., Khurana, D., Srinivasan, A.: SNARGs for P from sub-exponential DDH and QR. In: EUROCRYPT, pp. 520–549 (2022)

HW15. Hubacek, P., Wichs, D.: On the communication complexity of secure function evaluation with long output. In: ITCS, pp. 163–172 (2015)

KLVW23. Kalai, Y., Lombardi, A., Vaikuntanathan, V., Wichs, D.: Boosting batch arguments and RAM delegation. In: STOC, pp. 1545–1552 (2023)

KPY19. Kalai, Y.T., Paneth, O., Yang, L.: How to delegate computations publicly. In: STOC, pp. 1115–1124 (2019)

KVZ21. Kalai, Y.T., Vaikuntanathan, V., Zhang, R.Y.: Somewhere statistical soundness, post-quantum security, and SNARGs. In: TCC, pp. 330–368 (2021)

NWW23. Nassar, S., Waters, B., Wu, D.J.: Monotone policy Bargs from Bargs and additively homomorphic encryption. IACR Cryptol. ePrint Arch. (2023)
NY90. Naor, M., Yung, M.: 1 Public-key cryptosystems provably secure against chosen ciphertext attacks. In: STOC, pp. 427–437 (1990)
Pai99. Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. In: EUROCRYPT, pp. 223–238 (1999)
PP22. Paneth, O., Pass, R.: Incrementally verifiable computation via rate-1 batch arguments. In: FOCS, pp. 1045–1056 (2022)
PR17. Paneth, O., Rothblum, G.N.: On zero-testable homomorphic encryption and publicly verifiable non-interactive arguments. In: TCC, pp. 283–315 (2017)
Reg05. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: STOC, pp. 84–93 (2005)
Vad06. Vadhan, S.P.: An unconditional study of computational zero knowledge. SIAM J. Comput. **36**(4), 1160–1214 (2006)
WW22. Waters, B., Wu, D.J.: Batch arguments for NP and more from standard bilinear group assumptions. In: CRYPTO, pp. 433–463 (2022)

Batch Arguments to NIZKs from One-Way Functions

Eli Bradley[1(✉)], Brent Waters[1,2], and David J. Wu[1]

[1] University of Texas at Austin, Austin, TX, USA
elibradley@utexas.edu
[2] NTT Research, Sunnyvale, CA, USA

Abstract. Succinctness and zero-knowledge are two fundamental properties in the study of cryptographic proof systems. Several recent works have formalized the connections between these two notions by showing how to realize non-interactive zero-knowledge (NIZK) arguments from succinct non-interactive arguments. Specifically, Champion and Wu (CRYPTO 2023) as well as Bitansky, Kamath, Paneth, Rothblum, and Vasudevan (ePrint 2023) recently showed how to construct a NIZK argument for NP from a (somewhere-sound) non-interactive batch argument (BARG) and a dual-mode commitment scheme (and in the case of the Champion-Wu construction, a local pseudorandom generator). The main open question is whether a BARG suffices for a NIZK (just assuming one-way functions).

In this work, we first show that an *adaptively-sound* BARG for NP together with an one-way function imply a computational NIZK argument for NP. We then show that the weaker notion of somewhere soundness achieved by existing BARGs from standard algebraic assumptions are also *adaptively* sound if we assume *sub-exponential* security. This transformation may also be of independent interest. Taken together, we obtain a NIZK argument for NP from one-way functions and a sub-exponentially-secure somewhere-sound BARG for NP.

If we instead assume *plain* public-key encryption, we show that a standard *polynomially-secure* somewhere-sound batch argument for NP suffices for the same implication. As a corollary, this means a somewhere-sound BARG can be used to generically upgrade any semantically-secure public-key encryption scheme into one secure against chosen-ciphertext attacks. More broadly, our results demonstrate that constructing non-interactive batch arguments for NP is essentially no easier than constructing NIZK arguments for NP.

1 Introduction

A non-interactive argument system for an NP relation $\mathcal{R}$ allows a (computationally-bounded) prover to convince a verifier that a statement $x \in \{0,1\}^*$ is true (i.e., that $x \in \mathcal{L}$) with a single message π (which is referred to as a "proof"). The argument system is succinct if the size of the proof π is sublinear in the size of the circuit computing $\mathcal{R}$ and is zero-knowledge [21] if the

E. Boyle and M. Mahmoody (Eds.): TCC 2024, LNCS 15365, pp. 431–463, 2025.
https://doi.org/10.1007/978-3-031-78017-2_15

proof π reveals nothing more about x other than the fact that x is true. Both succinctness and zero-knowledge are fundamental properties of proof systems, and their combination in the form of zero-knowledge succinct non-interactive arguments (zkSNARGs) have found extensive applications to verifiable computation, authentication schemes, and privacy-preserving digital currencies. A recent line of works [3,11,26] have studied the formal relationship between succinctness and zero-knowledge. Since a succinct argument is not long enough to encode a traditional NP witness, it intuitively must lose some information about the witness associated with the statement. Thus, it is not surprising that succinct argument systems give rise to zero-knowledge arguments. Such a connection was first formalized by Kitagawa, Matsuda, and Yamakawa [26], who showed that a succinct non-interactive argument for NP can be used in conjunction with one-way functions to construct a non-interactive zero-knowledge (NIZK) argument.

Batch Arguments. The construction in [26] uses *adaptively-sound* SNARGs for NP as its starting point. However, constructing adaptively-sound SNARGs for NP in the plain model is challenging, and currently, all existing constructions from falsifiable assumptions rely on indistinguishability obfuscation [40–42]. The question then is whether argument systems satisfying *weaker* notions of succinctness could still imply zero-knowledge. This was studied recently in two works [3,11], which established a similar implication starting from the *weaker* notion of a non-interactive batch argument (BARG). Batch arguments allow a prover to amortize the communication cost of NP verification; namely, the prover can convince the verifier of a collection of t NP statements $(x_1, \ldots, x_t)$ with a proof of size $\mathsf{poly}(\lambda, s) \cdot o(t)$, where s is the size of the circuit computing the associated NP relation and λ is a security parameter. Unlike the case of SNARGs, a recent line of works have shown how to construct batch arguments for NP from a broad range of standard number-theoretic assumptions [8,12–14,16,18,23–25,33,39]. The work of Champion and Wu [11] showed how to obtain a computational NIZK argument from a somewhere-sound[1] BARG for NP in conjunction with a dual-mode commitment scheme and a (sub-exponentially-secure) local pseudorandom generator (PRG). The work of Bitansky, Kamath, Paneth, Rothblum, and Vasudevan [3] showed how to obtain a *statistical* NIZK argument from a somewhere-sound BARG for NP with a dual-mode commitment scheme.[2] In light of these works, a natural question is whether we can construct NIZK arguments solely from BARGs (and one-way functions).

[1] A BARG satisfies somewhere soundness if the common reference string (CRS) can be programmed at a specific index i, and adaptive soundness is guaranteed with respect to the i^{th} statements x_i. Moreover, the CRS (computationally) hides the special index i.

[2] In an independent update that was concurrent to this work, the most recent revision of their work [4] also show how to construct computational NIZK arguments from batch arguments and one-way functions. We discuss the concurrent work at the end of this section.

This Work. In this work, we first show how to construct a computational NIZK argument for NP from any one-way function together with a (polynomially-secure) *adaptively-sound* BARG for NP. Adaptive soundness is the most "natural" security notion for a BARG and can often be more convenient to use in constructions. However, most BARG constructions based on standard algebraic assumptions (e.g., [12,14,16,39]) only satisfy a weaker notion of "somewhere soundness" where the adversary has to pre-commit to the index of the false instance. As an additional contribution (of independent interest), we show in Sect. 5 that using complexity leveraging, any *sub-exponentially-secure* somewhere-sound BARG can also be used to obtain an adaptively-sound BARG. Thus, existing somewhere-sound BARGs based on standard algebraic assumptions also satisfy adaptive soundness at the expense of making a stronger sub-exponential hardness assumption. We believe this fact could also be useful in other applications of BARGs. In combination, we obtain a NIZK argument for NP from one-way functions and either (1) an adaptively-sound BARG for NP; *or* (2) a sub-exponentially-secure somewhere-sound BARG for NP (Corollary 1). Compared with [3,11], our construction only requires BARGs and one-way functions. The previous works [3,11] additionally relied on a dual-mode commitment scheme (and in the case of [11], also a local pseudorandom generator).

If we additionally assume *vanilla* public-key encryption, then we can obtain a computational NIZK for NP from a *polynomially-secure* somewhere-sound BARG for NP (Corollary 2). Namely, the use of public-key encryption allows us to relax the adaptive soundness requirement on the BARG to somewhere soundness. Like previous works, our constructions do *not* need extractability (although most existing BARGs support some type of extraction). We summarize our main results in the following informal theorem:

Theorem 1 (Informal). *There exists a NIZK for* NP *assuming the existence of* either

- *a one-way function and an* adaptively-sound *BARG for* NP*; or*
- *a public-key encryption scheme and a* somewhere-sound *BARG for* NP.

Moreover, adaptively-sound BARGs for NP *can be constructed from* sub-exponentially-secure *somewhere-sound BARGs for* NP.

Broadly speaking, our results demonstrate that constructing BARGs for NP is no easier than constructing (computational) NIZK arguments. Indeed, existing algebraic constructions of BARGs [12,14,16,39] are based on ideas and techniques that were previously used to build NIZK arguments.

An Implication to CCA-Security. Combined with classic results [32,36] on constructing public-key encryption with security against chosen-ciphertext attacks (CCA-security) from semantically-secure public-key encryption and (designated-verifier) NIZK arguments, our results show that BARGs for NP can be used to upgrade any semantically-secure public-key encryption scheme into a CCA-secure one. In this setting of upgrading the security of public-key encryption, we only require polynomial hardness of the BARG for NP (since we are given the semantically-secure public-key encryption to start).

Concurrent Work. In a recent and concurrent update[3] to their original work (December 2023) [4], Bitansky, Kamath, Paneth, Rothblum, and Vasudevan independently showed how to construct a computational NIZK argument for NP from a somewhere-sound non-interactive batch argument for NP and a one-way function. Notably, their construction only relies on polynomial-hardness of the underlying batch argument. The techniques in the two works are very different. We use a BARG to directly construct a hidden-bits generator (similar to [11,26]), which implies a NIZK via existing transformations [26,34]. On the other hand, [4] starts with a direct construction of a NIZK argument from a BARG which satisfies *distributional* zero-knowledge and has inverse polynomial zero-knowledge error. To obtain a full-fledged NIZK for NP with negligible zero-knowledge error, [4] takes a gate-by-gate approach followed by a privacy-amplification step (using multiparty computation techniques).

In another independent and concurrent work [30], Matsuda shows how to use a BARG to generically upgrade any CPA-secure public-key encryption scheme into a CCA-secure scheme. As noted above, such a transformation follows as an immediate corollary of Theorem 1. However, the approach from [30] can be instantiated with any BARG with proof size $t^{\varepsilon} \cdot \mathsf{poly}(\lambda, s)$, where t is the number of instances, s is the circuit size, and *any* constant $0 < \varepsilon < 1$. Our construction will rely on a BARG with proof size $t^{\varepsilon} \cdot \mathsf{poly}(\lambda, s)$ for *sufficiently small constant* ε. Many existing BARG constructions [12,14,16,18,23–25,39] achieve succinctness $\mathsf{poly}(\lambda, s, \log t)$, which satisfy both sets of requirements.

1.1 Technical Overview

Our construction follows the approach from [11,26] of using an (adaptively-sound) SNARG [26] or a (somewhere-sound) BARG [11] to construct a hidden-bits generator [34]. In conjunction with a NIZK in the ideal hidden bits model [17], this yields a NIZK in the common reference string (CRS) model. We start with a brief description of the hidden-bits model and the [11] construction, which is the starting point of this work.

The Hidden-Bits Model. The hidden-bits model [17] is an *idealized* model for constructing *unconditional* NIZK proofs. In this model, a trusted party samples a uniform *random* sequence of bits $r_1, \ldots, r_m \xleftarrow{R} \{0,1\}$. The trusted party gives $\mathbf{r} = r_1 r_2 \cdots r_m$ to the prover. To construct a proof for the statement x, the prover chooses a subset of indices $I \subseteq [m]$ and a proof string π. The trusted party then gives the bits $\mathbf{r}_I := \{r_i\}_{i \in I}$ and the proof string π to the verifier. The work [17]

[3] The original version of their work (May 2023) [5] showed how to construct a statistical NIZK argument with a *non-uniform* prover from somewhere-sound BARGs and lossy public-key encryption. In a subsequent revision (June 2023) [3], they improved their result to obtain a statistical NIZK argument with a *uniform* prover from somewhere-sound BARGs and lossy public-key encryption. In the most recent revision (December 2023) [4], they additionally showed how to obtain a computational NIZK argument from somewhere-sound BARGs. These works also show additional implications between batch arguments and (statistical) witness indistinguishability.

shows how to construct a NIZK for NP with statistical soundness and perfect zero-knowledge in the hidden-bits model.

Hidden-Bits Generators. Many works have shown how to leverage cryptographic tools to transform a NIZK in the hidden-bits model into a NIZK in the CRS model [2,9–11,17,20,26,28,34,37,38]. Similar to previous work [11,26], we use the abstraction based on the hidden-bits generators introduced by Quach, Rothblum, and Wichs [34]. Our presentation here is adapted from that in [11]. A hidden-bits generator allows a prover to generate a sequence of (pseudorandom) bits $\mathbf{r} = r_1 r_2 \cdots r_m$ and selectively reveal a subset of the bits $\mathbf{r}_I := \{r_i\}_{i \in I}$ for some $I \subseteq [m]$ to the verifier. Specifically, a hidden-bits generator consists of four algorithms $(\mathsf{Setup}, \mathsf{GenBits}, \mathsf{Prove}, \mathsf{Verify})$ with the following properties:

- The Setup algorithm takes the security parameter λ and the hidden-bits string length m, and outputs a common reference string crs for the hidden-bits generator.
- The $\mathsf{GenBits}$ algorithm takes the common reference string crs and outputs a hidden-bits string $\mathbf{r} \in \{0,1\}^m$ and a generator state st.
- The Prove algorithm takes the generator state st, a subset of indices $I \subseteq [m]$, and outputs a proof π for the subset of bits $\mathbf{r}_I := \{x_i\}_{i \in I}$.
- The Verify algorithm takes the common reference string crs, a subset of indices $I \subseteq [m]$, a subset of bits $\mathbf{r}_I = \{x_i\}_{i \in I}$, the opening π, and decides whether to accept or reject.

The correctness and security properties for a hidden-bits generator are defined as follows:

- **Correctness:** Correctness says that the verification algorithm accepts the proof output by Prove. Namely, if we sample $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^m)$ and $(\mathbf{r}, \mathsf{st}) \leftarrow \mathsf{GenBits}(\mathsf{crs})$, then for all $I \subseteq [m]$, $\mathsf{Verify}(\mathsf{crs}, I, \mathbf{r}_I, \mathsf{Prove}(\mathsf{st}, I)) = 1$.
- **Binding:** The binding property says that the set of valid hidden-bits strings (of length m) constitutes a *sparse* subset of the set of all m-bit strings. Namely, for every common reference string crs in the support of Setup, there exists a sparse subset $\mathcal{V}^{\mathsf{crs}} \subset \{0,1\}^m$ where $|\mathcal{V}^{\mathsf{crs}}| \leq 2^{m^\gamma \cdot \mathsf{poly}(\lambda)}$ for some constant $\gamma < 1$. Moreover, an efficient adversary can only come up with valid proofs π for sequences $\mathbf{r}_I \in \{0,1\}^{|I|}$ where $\mathbf{r}_I = \mathbf{r}'_I$ for some $\mathbf{r}' \in \mathcal{V}^{\mathsf{crs}}$.
- **Hiding:** The hiding property says that the unrevealed bits are pseudorandom. Specifically, for any set $I \subseteq [m]$ and sampling $\mathbf{r} \leftarrow \mathsf{GenBits}(\mathsf{crs})$, the distribution of the unrevealed bits $\mathbf{r}_{\bar{I}}$ (where $\bar{I} = [m] \setminus I$) is computationally indistinguishable from uniform given the common reference string crs, the revealed bits $\mathbf{r}_I$, and the proof π.

The Champion-Wu Hidden-Bits Generator. Building on the work of [26], Champion and Wu [11] recently showed how to construct a hidden-bits generator from a batch argument for NP, a (leakage-resilient) local PRG (i.e., a PRG where each output bit depends on a small number of bits of the seed, and the output remains pseudorandom when adversaries obtain some leakage on the PRG seed), and a

dual-mode commitment scheme. In a dual-mode commitment [15], the CRS can be sampled in one of two computationally indistinguishable modes. One mode yields equivocable (or statistically hiding) commitments while the other yields extractable (or statistically binding) commitments. A dual-mode commitment can be built from a *lossy* public-key encryption scheme [1]. We provide a basic outline of the [11] construction below:

- The common reference string consists of a CRS for a dual-mode commitment scheme and a CRS for a somewhere-sound BARG for NP.
- To sample a hidden-bits string, the prover samples a seed $\mathbf{s} \in \{0,1\}^n$ for the leakage-resilient local PRG.[4] The hidden-bits string $\mathbf{r} \in \{0,1\}^m$ is the output of $\mathbf{r} := \mathsf{PRG}(\mathbf{s})$. To open to a subset $I \subseteq [m]$, the prover does the following:
  - Commit to the PRG seed $\mathbf{s}$ using the dual-mode commitment. Let σ be the resulting commitment.
  - The prover constructs a BARG proof π that for each output index $i \in I$, the output bit r_i is consistent with the i^{th} bit of $\mathsf{PRG}(\mathbf{s})$, where $\mathbf{s}$ is the seed associated with the commitment σ.

  The proof consists of the commitment σ together with the BARG proof π.
- To check the opening, the verifier simply checks the BARG proof π (with respect to the committed seed σ).

To argue binding and hiding security, the analysis in [11] proceeds as follows:

- **Binding:** To argue binding, [11] programs the CRS for the dual-mode commitment to be extracting. If the local PRG has super-linear stretch, then the image of the PRG is a sparse subset of $\{0,1\}^m$. Moreover, somewhere-soundness of the BARG ensures that the prover can only open to bit-strings that are consistent with *some* seed (specifically, the seed $\mathbf{s}$ associated with the commitment σ). In the security proof, this latter step relies on the reduction being able efficiently extract the PRG seed $\mathbf{s}$ from the commitment σ. Namely, if there is an index i where r_i does not match the i^{th} bit of $\mathsf{PRG}(\mathbf{s})$, then the instance associated with the i^{th} output bit r_i in the BARG must be false. This is sufficient to setup a reduction to somewhere soundness of the BARG.
- **Hiding:** To argue that the scheme is hiding, [11] first switches the dual-mode commitments to be equivocating (i.e., in this case, the commitment σ completely hides the seed $\mathbf{s}$). Then, it treats the BARG proof π as "leakage" on the PRG seed $\mathbf{s}$. As long as π is much shorter than $\mathbf{s}$, they can appeal to leakage-resilience of the local PRG to argue that the unrevealed bits remain pseudorandom. Since the length of π scales with the size of the circuit that computes each bit of the PRG output, [11] requires that each output bit of the PRG be computed by a circuit that is significantly shorter than the length of the PRG seed. This is why they require the PRG to have small locality.

[4] Technically, the construction in [11] composes an arbitrary (sub-exponentially-secure) local PRG with a randomness extractor. Using the Gentry-Wichs leakage-simulation lemma [19] and assuming sub-exponential hardness of the local PRG, this yields a leakage-resilient local PRG. For ease of exposition, we describe their blueprint assuming a leakage-resilient local PRG.

Our Approach. In this work, we make two key modifications to the previous construction of [11] that eliminates the need for both the dual-mode commitment as well as the local PRG. We describe these two techniques below:

- **Removing locality by committing to internal wires.** As noted above, the [11] approach assumed a local PRG because they needed to ensure that the size of the circuit computing the PRG is much smaller than the length of the PRG seed. They do this because each instance of the BARG is associated with one of the output bits of the PRG. In this work, we take a different approach. Instead of just committing to the bits of the PRG seed $\mathbf{s}$ (as in [11]), we instead commit to *all* of the wires in the circuit computing $\mathsf{PRG}(\mathbf{s})$. The BARG is then used to check not only the validity of $\mathbf{r}_I$ where $\mathbf{r} := \mathsf{PRG}(\mathbf{s})$, but all of the internal gates in the computation of $\mathsf{PRG}(\mathbf{s})$. In this setting, each statement in the BARG only needs to check correct computation of an individual *gate* (with respect to the committed wire values). The size of the circuit depends only on the security parameter for the *commitment* scheme (which can be set independently of the seed length of the PRG). As such, we no longer need to assume locality of the PRG. In particular, we construct the leakage-resilient PRG from a leakage-resilient weak pseudorandom function (PRF), which can in turn be based solely on (polynomial-hard) one-way functions [22,35].
- **One-time dual-mode commitments.** A closer examination of the [11] construction shows that *one-time* equivocation suffices for the security analysis.[5] Specifically, one-time equivocation for a (bit) commitment scheme means that it is possible to jointly sample a common reference string along with a single commitment $\tilde{c}$ and openings $\tilde{\sigma}_0, \tilde{\sigma}_1$ of $\tilde{c}$ to the bits 0 and 1, respectively. One-time equivocable (bit) commitments follow from the classic bit-commitment scheme of Naor [31], which is based on one-way functions.

Using these techniques, we now obtain the following two instantiations (which correspond to the two main implications in Theorem 1):

- **A construction based on one-way functions.** While Naor's dual-mode bit-commitment scheme is either statistically binding or one-time equivocable, it does not *support* an efficient extracting mode (i.e., we do not have an efficient extraction algorithm that takes as input an extraction trapdoor and a commitment and outputs the committed value). Note that lack of extraction is not surprising since an extractable commitment would imply a public-key encryption scheme. In this work, we provide an alternative proof of binding that relies on *adaptive* soundness of the BARG (as opposed to somewhere soundness). We then show that using complexity leveraging, any sub-exponentially-secure somewhere-sound BARG is also adaptively sound. While simple, this latter transformation may also be of independent interest; we describe this in Sect. 5. Taken together, this yields a hidden-bits generator from any sub-exponentially-secure somewhere-sound BARG for NP, and

[5] As shown in [26,34], this suffices for single-theorem zero-knowledge, which can be boosted to multi-theorem zero-knowledge using the classic transformation of [17].

correspondingly, a NIZK for NP from the same assumption. We describe this construction in Sect. 3.

– **A construction based on public-key encryption.** By composing Naor's bit commitment scheme with a public-key encryption scheme, we also show how to construct a one-time dual-mode commitment with an efficient (trapdoor) extraction algorithm as well as a one-time equivocation mode. Similar ideas were used in a number of works studying CCA-secure encryption [27] and designated-verifier NIZKs [29]. Combined with the blueprint above, we obtain a hidden-bits generator from any polynomially-secure somewhere-sound BARG for NP and a semantically-secure public-key encryption scheme. This yields a NIZK for NP from the same set of assumptions. We describe this construction in Sect. 4.

2 Preliminaries

We write λ to denote the security parameter. For a positive integer $n \in \mathbb{N}$, we write $[n]$ to denote the set $\{1, \ldots, n\}$. We write $\mathsf{poly}(\lambda)$ to denote a fixed function that is $O(\lambda^c)$ for some $c \in \mathbb{N}$ and $\mathsf{negl}(\lambda)$ to denote a function that is $o(\lambda^{-c})$ for all $c \in \mathbb{N}$. We say an event occurs with overwhelming probability if its complement occurs with negligible probability. We say an algorithm is efficient if it runs in probabilistic polynomial time in the length of its input. For a finite set S, we write $x \xleftarrow{R} S$ to denote a uniform random draw from S. When $\mathcal{D}$ is a probability distribution, we write $x \leftarrow \mathcal{D}$ to denote a sample from $\mathcal{D}$. We say that two ensembles of distributions $\mathcal{D}_1 := \{\mathcal{D}_{1,\lambda}\}_{\lambda \in \mathbb{N}}$ and $\mathcal{D}_2 := \{\mathcal{D}_{2,\lambda}\}_{\lambda \in \mathbb{N}}$ are computationally indistinguishable if no efficient adversary can distinguish them except with $\mathsf{negl}(\lambda)$ probability. We say they are statistically indistinguishable if their statistical distance is negligible. Throughout this work, we consider security against non-uniform adversaries.

Basic Cryptographic Primitives. We now recall the definitions of some standard cryptographic primitives.

Definition 1 (Public-Key Encryption). *A public-key bit-encryption scheme with message space $\mathcal{M} = \{\mathcal{M}_\lambda\}_{\lambda \in \mathbb{N}}$ is a triple of efficient algorithms $\Pi_{\mathsf{PKE}} = (\mathsf{Setup}, \mathsf{Encrypt}, \mathsf{Decrypt})$ with the following properties:*

– $\mathsf{Setup}(1^\lambda) \to (\mathsf{pk}, \mathsf{sk})$*: On input the security parameter $\lambda \in \mathbb{N}$, the setup algorithm outputs a public key* pk *and secret key* sk*.*
– $\mathsf{Encrypt}(\mathsf{pk}, m) \to \mathsf{ct}$*: On input the public key* pk *and a message $m \in \mathcal{M}$, the encryption algorithm outputs a ciphertext* ct*.*
– $\mathsf{Decrypt}(\mathsf{sk}, \mathsf{ct}) \to m$*: On input the secret key* sk *and a ciphertext* ct*, the decryption algorithm outputs a message $m \in \mathcal{M}$.*

We require Π_{PKE} to satisfy the following properties:

– **Correctness:** *For all $\lambda \in \mathbb{N}$ and all messages $m \in \mathcal{M}$,*

$$\Pr\left[\mathsf{Decrypt}(\mathsf{sk}, \mathsf{ct}) = m : \begin{array}{l}(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Setup}(1^\lambda) \\ \mathsf{ct} \leftarrow \mathsf{Encrypt}(\mathsf{pk}, m)\end{array}\right] = 1.$$

– **Semantic security:** *For a security parameter λ and a bit $\beta \in \{0,1\}$, we define the semantic security game between an adversary $\mathcal{A}$ and a challenger as follows:*
  1. *The challenger samples a key pair $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Setup}(1^\lambda)$ and gives $(1^\lambda, \mathsf{pk})$ to $\mathcal{A}$.*
  2. *Algorithm $\mathcal{A}$ outputs two message $m_0, m_1 \in \mathcal{M}_\lambda$. The challenger replies with $\mathsf{ct}_\beta \leftarrow \mathsf{Encrypt}(\mathsf{pk}, m_\beta)$.*
  3. *Algorithm $\mathcal{A}$ outputs a bit $b' \in \{0,1\}$, which is the output of the experiment.*

  Then Π_{PKE} satisfies semantic security if for all efficient adversaries $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$,

$$|\Pr[b' = 1 \mid \beta = 0] - \Pr[b' = 1 \mid \beta = 1]| = \mathsf{negl}(\lambda).$$

Leakage-Resilient Weak PRF. We now introduce the definition of a leakage-resilient weak pseudorandom function (PRF). First, recall that the weak PRF security game asserts that the PRF evaluations on *random* inputs are pseudorandom. We say a weak PRF is leakage-resilient if this pseudorandomness property holds even if the adversary gets *arbitrary* leakage on the PRF key. In this work, we consider a definition where the adversary is first allowed to request (an arbitrary polynomial number of) random evaluations of the weak PRF before specifying its leakage function. We give the formal definition below. Our definition is adapted from that of [22].

Definition 2 (Leakage-Resilient Weak Pseudorandom Function [22, adapted]**).** *Let $\mathcal{Y}$ be a finite set. A leakage-resilient weak pseudorandom function with output space $\mathcal{Y}$ is a pair of efficient algorithms $\Pi_{\mathsf{LRwPRF}} = (\mathsf{Setup}, \mathsf{Eval})$ with the following syntax:*

– $\mathsf{Setup}(1^\lambda, 1^\ell) \to k$*: On input the security parameter $\lambda \in \mathbb{N}$ and a leakage parameter $\ell \in \mathbb{N}$, the setup algorithm outputs a key k. We assume that k implicitly contains the security parameter 1^λ, the leakage parameter 1^ℓ, and defines the domain $\mathcal{X}$ of the PRF. Let $\kappa = \kappa(\lambda, \ell)$ be the bit-length of the key k output by $\mathsf{Setup}(1^\lambda, 1^\ell)$.*
– $\mathsf{Eval}(k, x) \to y$*: On input a key k (which specifies the domain $\mathcal{X}$ of the PRF) and an input $x \in \mathcal{X}$, the evaluation algorithm outputs a value $y \in \mathcal{Y}$. The evaluation algorithm is deterministic.*

For a security parameter λ and bit $\beta \in \{0,1\}$, we define the leakage-resilient weak pseudorandomness game between an adversary $\mathcal{A}$ and a challenger as follows:

1. ***Pre-challenge evaluation queries:*** *On input the security parameter 1^λ, algorithm $\mathcal{A}$ starts by outputting the leakage parameter 1^ℓ and the number of pre-challenge queries 1^s it would like to make. The challenger samples a key $k \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell)$. Let $\mathcal{X}$ be the domain of the PRF associated with k. The challenger samples inputs $x_1, \ldots, x_s \xleftarrow{R} \mathcal{X}$ and replies to $\mathcal{A}$ with $\{(x_i, \mathsf{Eval}(k, x_i))\}_{i \in [s]}$.*

2. ***Leakage query:*** *Algorithm $\mathcal{A}$ now outputs a Boolean circuit* $\mathsf{leak}\colon \{0,1\}^{\kappa(\lambda,\ell)} \to \{0,1\}^{\ell}$. *The challenger responds with* $\mathsf{leak}(k)$ *to* $\mathcal{A}$.
3. ***Challenge queries:*** *Algorithm $\mathcal{A}$ then outputs the number of challenge queries 1^t it would like to make.*
   - *If $\beta = 0$, the challenger samples $x'_i \xleftarrow{R} \mathcal{X}$, $y_i \leftarrow \mathsf{Eval}(k, x'_i)$ for each $i \in [t]$.*
   - *If $\beta = 1$, the challenger samples $x'_i \xleftarrow{R} \mathcal{X}$, $y_i \xleftarrow{R} \mathcal{Y}$ for each $i \in [t]$.*

   The challenger gives $\{(x'_i, y_i)\}_{i\in[t]}$ to $\mathcal{A}$.
4. **Output:** *Algorithm $\mathcal{A}$ outputs a bit $b' \in \{0,1\}$.*

Finally, we say Π_{LRwPRF} satisfies leakage-resilient weak pseudorandomness if for all efficient adversaries $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ *such that for all $\lambda \in \mathbb{N}$,*

$$|\Pr[b' = 1 \mid \beta = 0] - \Pr[b' = 1 \mid \beta = 1]| = \mathsf{negl}(\lambda).$$

The work of [35] show how to construct a leakage-resilient weak PRF from one-way functions where the size of the key scales with $\ell \cdot \mathsf{poly}(\lambda)$ and ℓ is the leakage parameter. Technically, [35] show how to construct a leakage-resilient symmetric encryption scheme, but their construction implicitly uses a leakage-resilient weak PRF. For completeness, we provide the details and analysis of their construction (adapted to the setting of leakage-resilient weak PRFs) in the full version of this paper [7]. Below, we state the main conclusion:

Theorem 2 (Leakage-Resilient Weak PRF from One-Way Functions [35, adapted]**).** *Let λ be a security parameter and ℓ be a leakage parameter. Suppose $\rho \geq O(\lambda + \log \ell)$. Assuming the existence of one-way functions, there exists a leakage-resilient weak PRF with domain $\{0,1\}^{\rho}$, range $\{0,1\}$, and key length $(\ell + \lambda)\lambda$.*

One-Time Dual-Mode Bit Commitment. Next, we recall the notion of a one-time dual-mode bit commitment scheme. This is a bit commitment scheme where the common reference string can be sampled in one of two computationally indistinguishable modes: binding mode and equivocable mode. When the CRS is in binding mode, the commitment scheme is statistically binding. When the CRS is sampled in equivocable mode, the CRS sampling algorithm outputs an equivocable commitment $\tilde{c}$ together with two openings $\tilde{\sigma}_0, \tilde{\sigma}_1$ of $\tilde{c}$ to the bits 0 and 1, respectively. In other words, the special commitment $\tilde{c}$ is an equivocable commitment that can be efficiently opened to a 0 and a 1. We now give the formal definition. Naor's classic bit commitment scheme [31] based on one-way functions is a one-time dual-mode bit commitment scheme.

Definition 3 (One-Time Dual-Mode Bit Commitment). *A one-time dual-mode bit commitment is a tuple of efficient algorithms $\Pi_{\mathsf{BC}} =$* $(\mathsf{SetupBind}, \mathsf{SetupEquivocate}, \mathsf{Commit}, \mathsf{Verify})$ *with the following syntax:*

- $\mathsf{SetupBind}(1^{\lambda}) \to \mathsf{crs}$*: On input the security parameter λ, the setup algorithm for the binding mode outputs a common reference string* crs.

- $\mathsf{SetupEquivocate}(1^\lambda) \to (\mathsf{crs}, \tilde{c}, \tilde{\sigma}_0, \tilde{\sigma}_1)$*: On input the security parameter* λ*, the setup algorithm for the equivocating mode outputs a common reference string* crs *along with a commitment* $\tilde{c}$ *and openings* $\tilde{\sigma}_0, \tilde{\sigma}_1$*.*
- $\mathsf{Commit}(\mathsf{crs}, b) \to (c, \sigma)$*: On input the common reference string* crs *and a bit* $b \in \{0,1\}$*, the commit algorithm outputs a commitment* c *and an opening* σ*.*
- $\mathsf{Verify}(\mathsf{crs}, c, b, \sigma) \to \{0,1\}$*: On input the common reference string* crs*, a commitment* c*, a bit* $b \in \{0,1\}$*, and an opening* σ*, the verification algorithm outputs a bit* $b' \in \{0,1\}$*.*

We require Π_{BC} *to satisfy the following properties:*

- **Correctness:** *For all security parameters* $\lambda \in \mathbb{N}$*, all common reference strings* crs *in the support of either* $\mathsf{SetupBind}(1^\lambda)$ *or* $\mathsf{SetupEquivocate}(1^\lambda)$*, and all bits* $b \in \{0,1\}$*,*

$$\Pr\left[\mathsf{Verify}(\mathsf{crs}, c, b, \sigma) = 1 : (c, \sigma) \leftarrow \mathsf{Commit}(\mathsf{crs}, b)\right] = 1.$$

- **Mode indistinguishability:** *For a security parameter* $\lambda \in \mathbb{N}$ *and a bit* $\beta \in \{0,1\}$*, we define the mode indistinguishability game between an adversary* $\mathcal{A}$ *and a challenger as follows:*
  1. *If* $\beta = 0$*, the challenger samples* $\mathsf{crs} \leftarrow \mathsf{SetupBind}(1^\lambda)$*. If* $\beta = 1$*, the challenger samples* $(\mathsf{crs}, \tilde{c}, \tilde{\sigma}_0, \tilde{\sigma}_1) \leftarrow \mathsf{SetupEquivocate}(1^\lambda)$*. The challenger gives* $(1^\lambda, \mathsf{crs})$ *to* $\mathcal{A}$*.*
  2. *Algorithm* $\mathcal{A}$ *outputs a bit* $b \in \{0,1\}$*. The challenger gives* c, σ *to* $\mathcal{A}$*, where* (c, σ) *are computed as follows:*
     - *If* $\beta = 0$*,* $(c, \sigma) \leftarrow \mathsf{Commit}(\mathsf{crs}, b)$*.*
     - *If* $\beta = 1$*,* $(c, \sigma) \leftarrow (\tilde{c}, \tilde{\sigma}_b)$*.*
  3. *Algorithm* $\mathcal{A}$ *outputs a bit* $b' \in \{0,1\}$*.*

  The bit commitment scheme satisfies mode indistinguishability if for all efficient adversaries $\mathcal{A}$*, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for all* $\lambda \in \mathbb{N}$*,*

$$\left|\Pr\left[b' = 1 \mid \beta = 0\right] - \Pr\left[b' = 1 \mid \beta = 1\right]\right| = \mathsf{negl}(\lambda).$$

- **Statistical binding in binding mode:** *For all security parameters* $\lambda \in \mathbb{N}$ *and all (not necessarily efficient) adversaries* $\mathcal{A}$*,*

$$\Pr\left[\begin{matrix}\mathsf{Verify}(\mathsf{crs}, c, 0, \sigma_0) = 1 \\ = \mathsf{Verify}(\mathsf{crs}, c, 1, \sigma_1)\end{matrix} : \begin{matrix}\mathsf{crs} \leftarrow \mathsf{SetupBind}(1^\lambda) \\ (c, \sigma_0, \sigma_1) \leftarrow \mathcal{A}(1^\lambda, \mathsf{crs})\end{matrix}\right] = \mathsf{negl}(\lambda).$$

Theorem 3 (Bit Commitment from One-Way Functions [31, adapted]**).** *Assuming the existence of one-way functions, there exists a one-time dual-mode bit commitment scheme.*

2.1 Cryptographic Proof Systems

In this section, we recall the definition of a non-interactive zero-knowledge (NIZK) argument for NP as well as that of a non-interactive batch argument (BARG) for NP. We will often consider the language of Boolean circuit satisfiability, which we recall below:

Definition 4 (Boolean Circuit Satisfiability). *The language* $\mathcal{L}_{\mathsf{SAT}}$ *of Boolean circuit satisfiability consists of pairs* $(C, \mathbf{x})$ *of circuits* $C\colon \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$ *and inputs* $\mathbf{x} \in \{0,1\}^n$ *such that there exists* $\mathbf{w} \in \{0,1\}^h$ *where* $C(\mathbf{x}, \mathbf{w}) = 1$*:*

$$\mathcal{L}_{\mathsf{SAT}} = \left\{ (C, \mathbf{x}) : \begin{array}{c} C\colon \{0,1\}^n \times \{0,1\}^h \to \{0,1\}, \mathbf{x} \in \{0,1\}^n \\ \exists \mathbf{w} \in \{0,1\}^h : C(\mathbf{x}, \mathbf{w}) = 1 \end{array} \right\}.$$

Non-interactive Zero-Knowledge. We now recall the notion of a non-interactive zero-knowledge argument [6,21] for an arbitrary NP language.

Definition 5 (NIZK Argument for NP). *A non-interactive zero-knowledge argument for an* NP *relation* $\mathcal{R}$ *(with associated language* $\mathcal{L}$*) is a tuple of efficient algorithms* $\Pi_{\mathsf{NIZK}} = (\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify})$ *with the following syntax:*

- $\mathsf{Setup}(1^\lambda) \to \mathsf{crs}$*: On input the security parameter* $\lambda \in \mathbb{N}$*, the setup algorithm outputs a common reference string* crs*.*
- $\mathsf{Prove}(\mathsf{crs}, \mathbf{x}, \mathbf{w}) \to \pi$*: On input the common reference string* crs*, a statement* $\mathbf{x}$*, and a witness* $\mathbf{w}$*, the prove algorithm outputs a proof* π*.*
- $\mathsf{Verify}(\mathsf{crs}, \mathbf{x}, \pi) \to b$*: On input the common reference string* crs*, a statement* $\mathbf{x}$*, and a proof* π*, the verification algorithm outputs a bit* $b \in \{0,1\}$*.*

Moreover, Π_{NIZK} *should satisfy the following properties:*

- **Completeness:** *For all* $\lambda \in \mathbb{N}$ *and all* $(\mathbf{x}, \mathbf{w}) \in \mathcal{R}$*,*

$$\Pr\left[\mathsf{Verify}(\mathsf{crs}, \mathbf{x}, \pi) = 1 : \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda); \\ \pi \leftarrow \mathsf{Prove}(\mathsf{crs}, \mathbf{x}, \mathbf{w}) \end{array}\right] = 1.$$

- **Adaptive computational soundness:** *For all efficient adversaries* $\mathcal{A}$*, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for all* $\lambda \in \mathbb{N}$*,*

$$\Pr\left[\mathbf{x} \notin \mathcal{L} \wedge \mathsf{Verify}(\mathsf{crs}, \mathbf{x}, \pi) = 1 : \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda) \\ (\mathbf{x}, \pi) \leftarrow \mathcal{A}(1^\lambda, \mathsf{crs}) \end{array}\right] = \mathsf{negl}(\lambda).$$

- **Adaptive multi-theorem computational zero-knowledge:** *For every efficient adversary* $\mathcal{A}$*, there exists an efficient simulator* $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ *and a negligible function* $\mathsf{negl}(\cdot)$ *such that for all* $\lambda \in \mathbb{N}$ *and sampling* $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda)$ *and* $(\widetilde{\mathsf{crs}}, \mathsf{st}_{\mathcal{S}}) \leftarrow \mathcal{S}_1(1^\lambda)$*, we have that*

$$\left|\Pr\left[\mathcal{A}^{\mathcal{O}_0(\mathsf{crs},\cdot,\cdot)}(1^\lambda, \mathsf{crs}) = 1\right] - \Pr\left[\mathcal{A}^{\mathcal{O}_1(\mathsf{st}_{\mathcal{S}},\cdot,\cdot)}(1^\lambda, \widetilde{\mathsf{crs}}) = 1\right]\right| = \mathsf{negl}(\lambda),$$

and where the oracles $\mathcal{O}_0$ *and* $\mathcal{O}_1$ *are defined as follows:*

  - $\mathcal{O}_0(\mathsf{crs}, \mathbf{x}, \mathbf{w})$*: On input the common reference string* crs*, a statement* $\mathbf{x}$*, and a witness* $\mathbf{w}$*, the oracle outputs* $\perp$ *if* $(\mathbf{x}, \mathbf{w}) \notin \mathcal{R}$*. If* $(\mathbf{x}, \mathbf{w}) \in \mathcal{R}$*, it outputs* $\mathsf{Prove}(\mathsf{crs}, \mathbf{x}, \mathbf{w})$*.*
  - $\mathcal{O}_1(\mathsf{st}_{\mathcal{S}}, \mathbf{x}, \mathbf{w})$*: On input the simulator state* $\mathsf{st}_{\mathcal{S}}$*, a statement* $\mathbf{x}$ *and a witness* $\mathbf{w}$*, the oracle outputs* $\perp$ *if* $(\mathbf{x}, \mathbf{w}) \notin \mathcal{R}$*. If* $(\mathbf{x}, \mathbf{w}) \in \mathcal{R}$*, it outputs* $\mathcal{S}_2(\mathsf{st}_{\mathcal{S}}, \mathbf{x})$*.*

Non-interactive Batch Arguments. Next, we recall the definition of a non-interactive batch argument for the language of Boolean circuit satisfiability. We start by defining the standard notion of *adaptive* (computational) soundness, and then follow it with a relaxation called somewhere soundness [14,25].

Definition 6 (Batch Argument for NP [14, adapted]). *A non-interactive batch argument (BARG) for Boolean circuit satisfiability is a tuple of three efficient algorithms* $\Pi_{\mathsf{BARG}} = (\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify})$ *with the following syntax:*

- $\mathsf{Setup}(1^\lambda, 1^T, 1^s) \to \mathsf{crs}$*: On input the security parameter* $\lambda \in \mathbb{N}$*, a bound on the number of instances* $T \in \mathbb{N}$*, and a bound on the circuit size* $s \in \mathbb{N}$*, the setup algorithm outputs a common reference string* crs*.*
- $\mathsf{Prove}(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_t), (\mathbf{w}_1, \ldots, \mathbf{w}_t)) \to \pi$*: On input the common reference string* crs*, a Boolean circuit* $C\colon \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$*, statements* $\mathbf{x}_1, \ldots, \mathbf{x}_t \in \{0,1\}^n$*, and witnesses* $\mathbf{w}_1, \ldots, \mathbf{w}_t \in \{0,1\}^h$*, the prove algorithm outputs a proof* π*.*
- $\mathsf{Verify}(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_t), \pi) \to b$*: On input the common reference string* crs*, the Boolean circuit* $C\colon \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$*, statements* $\mathbf{x}_1, \ldots, \mathbf{x}_t \in \{0,1\}^n$ *and a proof* π*, the verification algorithm outputs a bit* $b \in \{0,1\}$*.*

Moreover, Π_{BARG} *should satisfy the following properties:*

- **Completeness:** *For all* $\lambda, T, s \in \mathbb{N}$*, all Boolean circuits* $C\colon \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$ *of size at most* s*, all* $t \le T$*, all statements* $\mathbf{x}_1, \ldots, \mathbf{x}_t \in \{0,1\}^n$*, and all witnesses* $\mathbf{w}_1, \ldots, \mathbf{w}_t \in \{0,1\}^h$ *where* $C(\mathbf{x}_i, \mathbf{w}_i) = 1$ *for all* $i \in [t]$*,*

$$\Pr\left[\mathsf{Verify}(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_t), \pi) = 1 : \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^T, 1^s); \\ \pi \leftarrow \mathsf{Prove}(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_t), (\mathbf{w}_1, \ldots, \mathbf{w}_t)) \end{array}\right] = 1.$$

- **Succinct proof size:** *There exists a polynomial* $\mathsf{poly}(\cdot)$ *such that for all* $\lambda, T, s \in \mathbb{N}$*, all* crs *in the support of* $\mathsf{Setup}(1^\lambda, 1^T, 1^s)$*, and all Boolean circuits* $C\colon \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$ *of size at most* s*, all* $t \le T$*, all statements* $\mathbf{x}_1, \ldots, \mathbf{x}_t \in \{0,1\}^n$*, and all witnesses* $\mathbf{w}_1, \ldots, \mathbf{w}_t \in \{0,1\}^h$*, the size of the proof* π *output by* $\mathsf{Prove}(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_t), (\mathbf{w}_1, \ldots, \mathbf{w}_t))$ *satisfies* $|\pi| \le \mathsf{poly}(\lambda + \log t + s)$*.*
- **Adaptive soundness:** *For a security parameter* λ*, we define the adaptive security game between an adversary* $\mathcal{A}$ *and a challenger as follows:*
  1. *On input the security parameter* 1^λ*, algorithm* $\mathcal{A}$ *starts by outputting the bound on the number of instances* 1^T *and the bound on the circuit size* 1^s*.*
  2. *The challenger samples a common reference string* $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^T, 1^s)$ *and gives* crs *to* $\mathcal{A}$*.*
  3. *Algorithm* $\mathcal{A}$ *outputs a Boolean circuit* $C\colon \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$ *of size at most* s*, statements* $(\mathbf{x}_1, \ldots, \mathbf{x}_t)$ *where* $\mathbf{x}_i \in \{0,1\}^n$ *for all* $i \in [t]$ *and where* $t \le T$*, and a proof* π*.*
  4. *The output of the experiment is* 1 *if* $\mathsf{Verify}(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_t), \pi) = 1$ *and for some* $i \in [t]$*,* $(C, \mathbf{x}_i) \notin \mathcal{L}_{\mathsf{SAT}}$*. Otherwise, the output is* 0*.*

Then Π_{BARG} satisfies adaptive security if for all efficient adversaries $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $\Pr[b = 1] = \mathsf{negl}(\lambda)$ in the adaptive security game.

Definition 7 (Somewhere-Sound Batch Argument for NP [14, adapted]). *A somewhere-sound non-interactive batch argument (BARG) for Boolean circuit satisfiability is a tuple of three efficient algorithms $\Pi_{\mathsf{BARG}} = (\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify})$. The Prove and Verify algorithms are defined exactly as in Definition 6, while the Setup algorithm now takes an additional index parameter:*

- $\mathsf{Setup}(1^\lambda, 1^T, 1^s, i) \to \mathsf{crs}$*: On input the security parameter $\lambda \in \mathbb{N}$, a bound on the number of instances $T \in \mathbb{N}$, a bound on the circuit size $s \in \mathbb{N}$, and an index $i \in [T]$, the setup algorithm outputs a common reference string* crs*.*

Moreover, the adaptive soundness property from Definition 6 is replaced by the following index hiding and somewhere soundness properties:

- **Index hiding:** *For a security parameter λ and a bit $\beta \in \{0,1\}$, we define the index-hiding game between an adversary $\mathcal{A}$ and a challenger as follows:*
  1. *On input the security parameter 1^λ, algorithm $\mathcal{A}$ starts by outputting the bound on the number of instances 1^T, the bound on the circuit size 1^s, and a pair of indices $i_0, i_1 \in [T]$.*
  2. *The challenger then proceeds to sample a common reference string $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^T, 1^s, i_\beta)$ and gives crs to $\mathcal{A}$.*
  3. *Algorithm $\mathcal{A}$ outputs a bit $b' \in \{0,1\}$.*

  Then Π_{BARG} satisfies index hiding if for all efficient adversaries $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$,

$$|\Pr[b' = 1 \mid \beta = 0] - \Pr[b' = 1 \mid \beta = 1]| = \mathsf{negl}(\lambda).$$

  We say Π_{BARG} satisfies sub-exponential index hiding security if there exists some constant $c > 1$ such that for all adversaries $\mathcal{A}$ running in time $2^{\lambda^{1/c}} \cdot \mathsf{poly}(\lambda)$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$,

$$|\Pr[b' = 1 \mid \beta = 0] - \Pr[b' = 1 \mid \beta = 1]| = \mathsf{negl}(\lambda).$$

- **Somewhere soundness:** *For a security parameter λ, we define the somewhere-soundness game between an adversary $\mathcal{A}$ and a challenger as follows:*
  1. *On input the security parameter 1^λ, algorithm $\mathcal{A}$ starts by outputting the bound on the number of instances 1^T, the bound on the circuit size 1^s, and the index $i^* \in [T]$.*
  2. *The challenger then proceeds to sample a common reference string $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^T, 1^s, i^*)$ and gives crs to $\mathcal{A}$.*
  3. *Algorithm $\mathcal{A}$ outputs a Boolean circuit $C\colon \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$ of size at most s, statements $(\mathbf{x}_1, \ldots, \mathbf{x}_t)$ where $\mathbf{x}_i \in \{0,1\}^n$ for all $i \in [t]$ and $t \leq T$, and a proof π.*
  4. *The output of the experiment is $b = 1$ if it is the case that $i^* \leq t$, $\mathsf{Verify}(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_t), \pi) = 1$, and $(C, \mathbf{x}_i^*) \notin \mathcal{L}_{\mathsf{SAT}}$. Otherwise, the output is $b = 0$.*

  Then Π_{BARG} satisfies somewhere-soundness if for all efficient adversaries $\mathcal{A}$, $\Pr[b = 1] = \mathsf{negl}(\lambda)$ in the somewhere-soundness game.

Constructions of BARGs. A number of recent works have established the existence of BARGs for NP from standard number-theoretic assumptions including the learning with errors (LWE) assumption [14,16], the k-Lin assumption over groups with bilinear maps [39], and (sub-exponential) hardness of decisional Diffie-Hellman (DDH) over pairing-free groups [12].

Soundness Definitions. Somewhere soundness is a relaxation of adaptive soundness where the adversary has the ability to choose the statements based on the CRS, but it is required to pre-commit to the index of a false statement. In Sect. 5 (Theorem 20), we show that using complexity leveraging, any somewhere-sound BARG which satisfies sub-exponential index hiding implies an adaptively-sound BARG. Namely, if index hiding security of the somewhere-sound BARG holds against an adversary that is able to decide the underlying NP language, then somewhere soundness implies adaptive soundness by a simple guessing argument. As a security notion, adaptive soundness is sometimes more convenient to work with compared to somewhere soundness. As such, we use adaptive soundness in our one-way function based construction in Sect. 3, which yields a construction based on sub-exponentially secure somewhere-sound BARGs. For our construction based on public-key encryption (Sect. 4), somewhere soundness suffices for our security analysis and we avoid the need for complexity leveraging. This yields a construction based only on polynomial hardness, but additionally relies on public-key encryption.

2.2 Hidden-Bits Generator

In this section, we recall the notion of a hidden-bits generator with subset-dependent proofs from [26,34]. For a bitstring $\mathbf{r} \in \{0,1\}^n$ and a set of indices $I \subseteq [n]$, we write $\mathbf{r}_I \in \{0,1\}^{|I|}$ to denote the substring corresponding to the bits of $\mathbf{r}$ indexed by I. Our presentation here is adapted from the work of [11].

Definition 8 (Hidden-Bits Generator [26, Definition 11]). *A hidden-bits generator with subset-dependent proofs is a tuple of efficient algorithms* $\Pi_{\mathsf{HBG}} = (\mathsf{Setup}, \mathsf{GenBits}, \mathsf{Prove}, \mathsf{Verify})$ *with the following syntax:*

- $\mathsf{Setup}(1^\lambda, 1^m) \to \mathsf{crs}$*: On input the security parameter* λ*, and the output length* m*, the setup algorithm outputs a common reference string* crs*.*
- $\mathsf{GenBits}(\mathsf{crs}) \to (\mathbf{r}, \mathsf{st})$*: On input the the common reference string* crs*, the generator algorithm outputs a string* $\mathbf{r} \in \{0,1\}^m$ *and a state* st*.*
- $\mathsf{Prove}(\mathsf{st}, I) \to \pi$*: On input the state* st *and a subset* $I \subseteq [m]$*, the prove algorithm outputs a proof* π*.*
- $\mathsf{Verify}(\mathsf{crs}, I, \mathbf{r}_I, \pi) \to b$*: On input a common reference string* crs*, a subset* $I \subseteq [m]$*, a string* $\mathbf{r}_I \in \{0,1\}^{|I|}$*, and a proof* π*, the verification algorithm outputs a bit* $b \in \{0,1\}$*.*

We require Π_{HBG} *to satisfy the following properties:*

– ***Correctness:*** *For all $\lambda, m \in \mathbb{N}$ and all subsets $I \subseteq [m]$, we have*

$$\Pr\left[\mathsf{Verify}(\mathsf{crs}, I, \mathbf{r}_I, \pi) = 1 : \begin{array}{r}\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^m);\\ (\mathbf{r}, \mathsf{st}) \leftarrow \mathsf{GenBits}(\mathsf{crs});\\ \pi \leftarrow \mathsf{Prove}(\mathsf{st}, I)\end{array}\right] = 1.$$

– ***Somewhat computational binding:*** *For every* crs *in the support of the algorithm* $\mathsf{Setup}(1^\lambda, 1^m)$*, there exists a set* $\mathcal{V}^{\mathsf{crs}}$ *with the following properties:*

(i) **Output sparsity.** *There exists a universal constant $\gamma < 1$ and a fixed polynomial $p(\cdot)$ such that for every polynomial $m = m(\lambda)$, there exists $\lambda_m \in \mathbb{N}$ such that for all $\lambda \geq \lambda_m$ and every* crs *in the support of* $\mathsf{Setup}(1^\lambda, 1^m)$*, $|\mathcal{V}^{\mathsf{crs}}| \leq 2^{m^\gamma \cdot p(\lambda)}$*

(ii) **Computational binding.** *For a security parameter λ, we define the computational binding game between an adversary $\mathcal{A}$ and a challenger as follows:*

(a) On input the security parameter 1^λ, algorithm $\mathcal{A}$ starts by outputting the output length 1^m.

(b) The challenger samples $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^m)$ *and gives* crs *to $\mathcal{A}$.*

(c) Algorithm $\mathcal{A}$ outputs a tuple $(I, \mathbf{r}_I, \pi)$.

(d) The output of the experiment is $b = 1$ if it holds that $\mathbf{r}_I \notin \mathcal{V}_I^{\mathsf{crs}}$ and $\mathsf{Verify}(\mathsf{crs}, I, \mathbf{r}_I, \pi) = 1$*, where $\mathcal{V}_I^{\mathsf{crs}} := \{\mathbf{r}_I : \mathbf{r} \in \mathcal{V}^{\mathsf{crs}}\}$. Otherwise, the output is $b = 0$.*

We say the Π_{HBG} is computationally binding if for all efficient adversaries $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ *such that for all $\lambda \in \mathbb{N}$,* $\Pr[b = 1] = \mathsf{negl}(\lambda)$ *in the computational binding security game.*

– ***Computational hiding:*** *For a security parameter λ and bit $\beta \in \{0, 1\}$, we define the computational hiding game between an adversary $\mathcal{A}$ and a challenger as follows:*

1. *On input the security parameter 1^λ, algorithm $\mathcal{A}$ starts by outputting the output length 1^m and a subset $I \subseteq [m]$.*
2. *The challenger samples* $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^m)$*,* $(\mathbf{r}, \mathsf{st}) \leftarrow \mathsf{GenBits}(\mathsf{crs})$*,* $\pi \leftarrow \mathsf{Prove}(\mathsf{st}, I)$ *and* $\mathbf{r}' \xleftarrow{R} \{0,1\}^m$*.*
   - *If $\beta = 0$, the challenger gives $(\mathsf{crs}, I, \mathbf{r}_I, \pi, \mathbf{r}_{\bar{I}})$ to $\mathcal{A}$, where $\bar{I} = [m] \setminus I$.*
   - *If $\beta = 1$, the challenger gives $(\mathsf{crs}, I, \mathbf{r}_I, \pi, \mathbf{r}'_{\bar{I}})$ to $\mathcal{A}$ where $\bar{I} = [m] \setminus I$.*
3. *Algorithm $\mathcal{A}$ outputs a bit $b' \in \{0, 1\}$.*

We say the Π_{HBG} is computationally hiding if for all efficient adversaries $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ *such that for all $\lambda \in \mathbb{N}$,*

$$|\Pr[b' = 1 \mid \beta = 1] - \Pr[b' = 1 \mid \beta = 0]| = \mathsf{negl}(\lambda).$$

Theorem 4 (NIZK from Hidden-Bits Generator [26]). *If there exists a hidden-bits generator with subset-dependent proofs, then there exists a computational NIZK argument for* NP*. The NIZK argument satisfies adaptive computational soundness and adaptive multi-theorem zero knowledge.*

3 Hidden-Bits Generator from Adaptively-Sound BARGs and OWFs

In this section, we show how to construct a hidden-bits generator with subset-dependent proofs using an adaptively-sound batch argument for NP, a one-time dual-mode bit commitment scheme, and a leakage-resilient weak PRF. By Theorems 3 and 2, the dual-mode bit commitment scheme and the leakage-resilient weak PRF can be constructed from one-way functions. From Theorem 20, we can construct an adaptively-sound BARG for NP from any sub-exponentially-secure somewhere-sound BARG for NP. Invoking Theorem 4 now yields a NIZK for NP from any sub-exponentially-secure somewhere-sound BARG for NP (Corollary 1).

Boolean Circuits. We start by describing the conventions we use for describing Boolean circuits. Let $C\colon \{0,1\}^n \to \{0,1\}$ be a Boolean circuit where each gate is a fan-in-2 NAND gate. Let s be the size of C, as measured by the number of wires in C. We associate an index $i \in [s]$ with each wire:

- **Input wires:** We index the n input wires with the values $1, \ldots, n$ and will refer to the wire at index $i \in [n]$ as the "i^{th} input wire" to C.
- **Internal wires:** Each non-input wire is associated with an index $i \in \{n+1, \ldots, s\}$ with the property that the value of wire i is completely determined by the value of the wires indexed $j_{i,\mathrm{L}}, j_{i,\mathrm{R}} \in \{1, \ldots, i-1\}$. Specifically, the value of wire i is the NAND of the value of its left input wire (i.e., the wire indexed $j_{i,\mathrm{L}}$) and the value of its right input wire (i.e., the wire indexed $j_{i,\mathrm{R}}$).
- **Output wire:** The output wire is associated with the index s.

Construction 5 (Hidden-Bits Generator from Batch Arguments). Let $\lambda \in \mathbb{N}$ be a security parameter and $m \in \mathbb{N}$ be an output length parameter. Our construction depends on the following primitives:

- Let $\Pi_{\mathsf{LRwPRF}} = (\mathsf{LRwPRF.Setup}, \mathsf{LRwPRF.Eval})$ be a leakage-resilient weak PRF (Definition 2) with range $\{0,1\}$. Let $\ell = \ell(\lambda, m)$ be a leakage parameter which will be set according to the requirements of the security analysis (Theorems 7 and 8). Let $\kappa = \kappa(\lambda, m)$ be the bit-length of the keys output by $\mathsf{LRwPRF.Setup}(1^\lambda, 1^{\ell(\lambda,m)})$. Let $n = n(\lambda, m)$ be the bit-length of the domain of LRwPRF (when instantiated with security parameter λ and leakage parameter $\ell = \ell(\lambda, m)$).
- Let $C\colon \{0,1\}^{\kappa+n} \to \{0,1\}$ be the Boolean circuit that evaluates Π_{LRwPRF}. Namely, $C(k, \mathbf{z}) := \mathsf{LRwPRF.Eval}(k, \mathbf{z})$. Let s be the size of C (i.e., the number of wires in C). In the following, we will define the following sets to refer to the wires in C:
  - Let $S_{\mathsf{key}} = \{1, \ldots, \kappa\}$ be the indices of the wires corresponding to the PRF key.
  - Let $S_{\mathsf{eval}} = \{\kappa+1, \ldots, \kappa+n\}$ be the indices of the wires corresponding to the evaluation point $\mathbf{z}$.
  - Let $S_{\mathsf{int}} = \{\kappa+n+1, \ldots, s\}$ be the indices of the non-input wires.

- Let $\Pi_{\mathsf{BC}} = (\mathsf{BC.SetupBind}, \mathsf{BC.SetupEquivocate}, \mathsf{BC.Commit}, \mathsf{BC.Verify})$ be a one-time dual-mode bit commitment scheme (Definition 3).
- Let $\Pi_{\mathsf{BARG}} = (\mathsf{BARG.Setup}, \mathsf{BARG.Prove}, \mathsf{BARG.Verify})$ be an (adaptively-sound) batch argument for NP. Let $\ell_{\mathsf{BARG}} = \ell_{\mathsf{BARG}}(\lambda, T, s)$ denote a bound on the length of the proof output by Π_{BARG} as a function of the bound on the number of instances T and the size s of the associated NP relation.
- Define the NP relation $\mathcal{R}$ as follows:

> **Statement:** common reference string $\mathsf{crs}_{\mathsf{BC}}^{(\mathrm{L})}, \mathsf{crs}_{\mathsf{BC}}^{(\mathrm{R})}, \mathsf{crs}_{\mathsf{BC}}^{(\mathrm{OUT})}$ and commitments $c_{\mathrm{L}}, c_{\mathrm{R}}, c_{\mathrm{OUT}}$
> **Witness:** bits $b_{\mathrm{L}}, b_{\mathrm{R}}, b_{\mathrm{OUT}} \in \{0,1\}$ and openings $\sigma_{\mathrm{L}}, \sigma_{\mathrm{R}}, \sigma_{\mathrm{OUT}}$
> Output 1 if the following conditions hold:
>
> - For each $i \in \{\mathrm{L}, \mathrm{R}, \mathrm{OUT}\}$, $\mathsf{BC.Verify}(\mathsf{crs}_{\mathsf{BC}}^{(i)}, c_i, b_i, \sigma_i) = 1$.
> - $b_{\mathrm{OUT}} = \mathrm{NAND}(b_{\mathrm{L}}, b_{\mathrm{R}})$.
>
> Otherwise, output 0.

Let $C_{\mathcal{R}}$ be the circuit computing the NP relation $\mathcal{R}$ and $\mathcal{L}$ be the associated NP language.

We construct our hidden-bits generator $\Pi_{\mathsf{HBG}} = (\mathsf{Setup}, \mathsf{GenBits}, \mathsf{Prove}, \mathsf{Verify})$ as follows:

- $\mathsf{Setup}(1^\lambda, 1^m)$: On input the security parameter λ and output length m, the setup algorithm start by sampling the following collection of common reference strings for the bit commitment scheme:
  - **CRS for the key:** For $j \in S_{\mathsf{key}}$, sample $\mathsf{crs}_{\mathsf{BC}}^{(\mathsf{key},j)} \leftarrow \mathsf{BC.SetupBind}(1^\lambda)$.
  - **CRS for the evaluation point:** For $i \in [m]$ and $j \in S_{\mathsf{eval}}$, sample $\mathsf{crs}_{\mathsf{BC}}^{(i,j)} \leftarrow \mathsf{BC.SetupBind}(1^\lambda)$.
  - **CRS for the non-input wires:** For $i \in [m]$ and $j \in S_{\mathsf{int}}$, sample $\mathsf{crs}_{\mathsf{BC}}^{(i,j)} \leftarrow \mathsf{BC.SetupBind}(1^\lambda)$.

  Next, the setup algorithm samples $\mathbf{z}_1, \ldots, \mathbf{z}_m \xleftarrow{\mathrm{R}} \{0,1\}^n$. The setup algorithm commits to $\mathbf{z}_1, \ldots, \mathbf{z}_m$ as follows:
  - For each $i \in [m]$ and $j \in S_{\mathsf{eval}}$, compute the commitments and openings $(c^{(i,j)}, \sigma^{(i,j)}) \leftarrow \mathsf{BC.Commit}\big(\mathsf{crs}_{\mathsf{BC}}^{(i,j)}, z_{i,j-\kappa}\big)$.

  Let $s_{\mathsf{int}} = |S_{\mathsf{int}}| = s - (\kappa + n)$ be the number of non-input wires in C. Sample a CRS for the BARG: $\mathsf{crs}_{\mathsf{BARG}} \leftarrow \mathsf{BARG.Setup}(1^\lambda, 1^{m s_{\mathsf{int}}}, 1^{|C_{\mathcal{R}}|})$. Output the common reference string

$$\mathsf{crs} = \Big((\mathbf{z}_1, \ldots, \mathbf{z}_m), \big\{\mathsf{crs}_{\mathsf{BC}}^{(\mathsf{key},j)}\big\}_{j \in S_{\mathsf{key}}}, \big\{\mathsf{crs}_{\mathsf{BC}}^{(i,j)}\big\}_{i \in [m], j \in S_{\mathsf{eval}} \cup S_{\mathsf{int}}}, \big\{\big(c^{(i,j)}, \sigma^{(i,j)}\big)\big\}_{i \in [m], j \in S_{\mathsf{eval}}}, \mathsf{crs}_{\mathsf{BARG}}\Big). \quad (3.1)$$

– $\mathsf{GenBits}(\mathsf{crs})$: On input the common reference string crs (parsed according to Eq. (3.1)), the generator algorithm samples a weak PRF key $k \leftarrow \mathsf{LRwPRF.Setup}(1^\lambda, 1^\ell)$ and computes $r_i \leftarrow \mathsf{LRwPRF.Eval}(k, \mathbf{z}_i)$ for each $i \in [m]$. It outputs the hidden bits string $\mathbf{r} = r_1 \| \cdots \| r_m$ and the state $\mathsf{st} = (\mathsf{crs}, k)$.

– $\mathsf{Prove}(\mathsf{st}, I)$: On input the state $\mathsf{st} = (\mathsf{crs}, k)$ (where crs is parsed according to Eq. (3.1) and $k \in \{0,1\}^\kappa$) and a set of indices $I \subseteq [m]$, the prove algorithm proceeds as follows:

  - **Commit to the bits of k:** For each $j \in S_{\mathsf{key}}$, compute $\left(c^{(\mathsf{key},j)}, \sigma^{(\mathsf{key},j)}\right) \leftarrow \mathsf{BC.Commit}\left(\mathsf{crs}^{(\mathsf{key},j)}, k_j\right)$.
  - **Commit to the non-input wires for $C(k, \mathbf{z}_i)$:** For each $i \in [m]$, let $w_1^{(i)}, \ldots, w_s^{(i)}$ be the wire values of $C(k, \mathbf{z}_i)$. For each $i \in [m]$ and $j \in S_{\mathsf{int}}$, compute $\left(c^{(i,j)}, \sigma^{(i,j)}\right) \leftarrow \mathsf{BC.Commit}\left(\mathsf{crs}^{(i,j)}, w_j^{(i)}\right)$.
  - **Construct a BARG proof of validity:** Recall that $s_{\mathsf{int}} = |S_{\mathsf{int}}| = s - (\kappa + n)$ is the number of non-input wires in the circuit C. These indices associated with these wires are $\kappa + n + 1, \ldots, \kappa + n + s_{\mathsf{int}}$. Now, for each $i \in [m]$, define the following:
    * As before, let $w_1^{(i)}, \ldots, w_s^{(i)} \in \{0,1\}$ be the wire values of $C(k, \mathbf{z}_i)$.
    * For each $j \in [s_{\mathsf{int}}]$, let $j_{\mathrm{L}}, j_{\mathrm{R}}$ be the wire indices that determine the value of the j^{th} non-input wire $j_{\mathrm{OUT}} = (\kappa+n)+j$. Define the statement $x^{(i,j)}$ and witness $w^{(i,j)}$ as follows:

$$x^{(i,j)} = \left(\mathsf{crs}_{\mathsf{BC}}^{(i,j_{\mathrm{L}})}, \mathsf{crs}_{\mathsf{BC}}^{(i,j_{\mathrm{R}})}, \mathsf{crs}_{\mathsf{BC}}^{(i,j_{\mathrm{OUT}})}, c^{(i,j_{\mathrm{L}})}, c^{(i,j_{\mathrm{R}})}, c^{(i,j_{\mathrm{OUT}})}\right) \tag{3.2}$$

$$w^{(i,j)} = \left(w_{j_{\mathrm{L}}}^{(i)}, w_{j_{\mathrm{R}}}^{(i)}, w_{j_{\mathrm{OUT}}}^{(i)}, \sigma^{(i,j_{\mathrm{L}})}, \sigma^{(i,j_{\mathrm{R}})}, \sigma^{(i,j_{\mathrm{OUT}})}\right). \tag{3.3}$$

    Here, for all $i \in [m]$ and $j \in S_{\mathsf{key}}$, we adopt the convention that $\mathsf{crs}_{\mathsf{BC}}^{(i,j)} := \mathsf{crs}_{\mathsf{BC}}^{(\mathsf{key},j)}$, $c^{(i,j)} := c^{(\mathsf{key},j)}$, and $\sigma^{(i,j)} := \sigma^{(\mathsf{key},j)}$.

  Construct the proof

$$\pi_{\mathsf{BARG}} \leftarrow \mathsf{BARG.Prove}\left(\mathsf{crs}_{\mathsf{BARG}}, C_{\mathcal{R}}, (x^{(i,j)})_{i \in I, j \in [s_{\mathsf{int}}]}, (w^{(i,j)})_{i \in I, j \in [s_{\mathsf{int}}]}\right). \tag{3.4}$$

  Output

$$\pi = \left(\pi_{\mathsf{BARG}}, \{c^{(\mathsf{key},j)}\}_{j \in S_{\mathsf{key}}}, \{c^{(i,j)}\}_{i \in I, j \in S_{\mathsf{int}}}, \{\sigma^{(i,s)}\}_{i \in I}\right). \tag{3.5}$$

– $\mathsf{Verify}(\mathsf{crs}, I, \mathbf{r}_I, \pi)$: On input a common reference string crs (parsed according to Eq. (3.1)), a subset $I \subseteq [m]$, a string $\mathbf{r}_I \in \{0,1\}^{|I|}$, and a proof $\pi = \left(\pi_{\mathsf{BARG}}, \{c^{(\mathsf{key},j)}\}_{j \in S_{\mathsf{key}}}, \{c^{(i,j)}\}_{i \in I, j \in S_{\mathsf{int}}}, \{\sigma^{(i,s)}\}_{i \in I}\right)$, the verification algorithm checks the following conditions:

  - **Validity of output commitments:** For all $i \in I$, check that the output commitments satisfy $\mathsf{BC.Verify}\left(\mathsf{crs}_{\mathsf{BC}}^{(i,s)}, c^{(i,s)}, r_i, \sigma^{(i,s)}\right) = 1$.
  - **Validity of BARG proof:** For each $i \in I$ and $j \in [s_{\mathsf{int}}]$, compute $x^{(i,j)}$ from crs and π according to Eq. (3.2). Then, check that the BARG proof satisfies $\mathsf{BARG.Verify}(\mathsf{crs}_{\mathsf{BARG}}, C_{\mathcal{R}}, (x^{(i,j)})_{i \in I, j \in [s_{\mathsf{int}}]}, \pi_{\mathsf{BARG}}) = 1$.

  If both checks pass, output 1. Otherwise, output 0.

Theorem 6 (Correctness). *If Π_{BARG} is complete and Π_{BC} is correct, then Construction 5 is correct.*

Proof. Let λ be a security parameter, m be an output length, and $I \subseteq [m]$ a set of indices. Let $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^m)$, parsed according to Eq. (3.1). Let $(\mathbf{r}, \mathsf{st}) \leftarrow \mathsf{GenBits}(\mathsf{crs})$ where $\mathsf{st} = (\mathsf{crs}, k)$ and $\pi \leftarrow \mathsf{Prove}(\mathsf{st}, I)$. Then,

$$\pi = \left(\pi_{\mathsf{BARG}}, \{c^{(\mathsf{key},j)}\}_{j \in S_{\mathsf{key}}}, \{c^{(i,j)}\}_{i \in I, j \in S_{\mathsf{int}}}, \{\sigma^{(i,s)}\}_{i \in I}\right).$$

For each $i \in I$, let $w_1^{(i)}, \ldots, w_s^{(i)} \in \{0,1\}$ be the wire values of $C(k, \mathbf{z}_i)$. Consider the output of $\mathsf{Verify}(\mathsf{crs}, I, \mathbf{r}_I, \pi)$. First, we show that for every $i \in I$ and $j \in [s]$,

$$\mathsf{BC.Verify}\left(\mathsf{crs}_{\mathsf{BC}}^{(i,j)}, c^{(i,j)}, w_j^{(i)}, \sigma^{(i,j)}\right) = 1. \tag{3.6}$$

We consider three cases:

- Suppose $j \in S_{\mathsf{key}}$. By definition, this means $\mathsf{crs}_{\mathsf{BC}}^{(i,j)} = \mathsf{crs}_{\mathsf{BC}}^{(\mathsf{key},j)}$, $c^{(i,j)} = c^{(\mathsf{key},j)}$, $\sigma^{(i,j)} = \sigma^{(\mathsf{key},j)}$ and $w_j^{(i)} = k_j$. By construction, the Prove algorithm computes $(c^{(\mathsf{key},j)}, \sigma^{(\mathsf{key},j)}) \leftarrow \mathsf{BC.Commit}(\mathsf{crs}_{\mathsf{BC}}^{(\mathsf{key},j)}, k_j)$, where the Setup algorithm sampled $\mathsf{crs}_{\mathsf{BC}}^{(\mathsf{key},j)} \leftarrow \mathsf{BC.SetupBind}(1^\lambda)$. By correctness of Π_{BC},

$$\begin{aligned}\mathsf{BC.Verify}\left(\mathsf{crs}_{\mathsf{BC}}^{(i,j)}, c^{(i,j)}, w_j^{(i)}, \sigma^{(i,j)}\right) &= \mathsf{BC.Verify}\left(\mathsf{crs}_{\mathsf{BC}}^{(\mathsf{key},j)}, c^{(\mathsf{key},j)}, k_j, \sigma^{(\mathsf{key},j)}\right) \\ &= 1.\end{aligned}$$

- Suppose $j \in S_{\mathsf{eval}}$. Then, $w_j^{(i)} = z_{i,j-\kappa}$. By construction, the Setup algorithm samples $(c^{(i,j)}, \sigma^{(i,j)}) \leftarrow \mathsf{BC.Commit}(\mathsf{crs}_{\mathsf{BC}}^{(i,j)}, z_{i,j-\kappa})$. Again, since $\mathsf{crs}_{\mathsf{BC}}^{(i,j)} \leftarrow \mathsf{BC.SetupBind}(1^\lambda)$, it follows by correctness of Π_{BC} that

$$\begin{aligned}\mathsf{BC.Verify}\left(\mathsf{crs}_{\mathsf{BC}}^{(i,j)}, c^{(i,j)}, w_j^{(i)}, \sigma^{(i,j)}\right) &= \mathsf{BC.Verify}\left(\mathsf{crs}_{\mathsf{BC}}^{(i,j)}, c^{(i,j)}, z_{i,j-\kappa}, \sigma^{(i,j)}\right) \\ &= 1.\end{aligned}$$

- Suppose $j \in S_{\mathsf{int}}$. By construction, the Prove algorithm computes the commitments and openings $(c^{(i,j)}, \sigma^{(i,j)}) \leftarrow \mathsf{BC.Commit}(\mathsf{crs}^{(i,j)}, w_j^{(i)})$. Since $\mathsf{crs}_{\mathsf{BC}}^{(i,j)} \leftarrow \mathsf{BC.SetupBind}(1^\lambda)$, by correctness of Π_{BC},

$$\mathsf{BC.Verify}\left(\mathsf{crs}_{\mathsf{BC}}^{(i,j)}, c^{(i,j)}, w_j^{(i)}, \sigma^{(i,j)}\right) = 1.$$

We now consider the two checks performed by Verify:

- **Validity of output commitments.** Let $i \in I$. The value of the output wire is $w_s^{(i)} = C(k, \mathbf{z}_i) = r_i$. From Eq. (3.6),

$$\mathsf{BC.Verify}\left(\mathsf{crs}_{\mathsf{BC}}^{(i,s)}, c^{(i,s)}, r_i, \sigma^{(i,s)}\right) = \mathsf{BC.Verify}\left(\mathsf{crs}_{\mathsf{BC}}^{(i,s)}, c^{(i,s)}, w_s^{(i)}, \sigma^{(i,s)}\right) = 1.$$

– **Validity of BARG proof.** Take any index $i \in I$ and $j \in [s_{\mathsf{int}}]$. Let j_{L} and j_{R} be the wire indices that determine the value of the j^{th} non-input wire $j_{\mathrm{OUT}} = (\kappa + n) + j$. Let $x^{(i,j)}$ and $w^{(i,j)}$ be the statement and witness as defined in Eqs. (3.2) and (3.3):

$$x^{(i,j)} = \left(\mathsf{crs}_{\mathsf{BC}}^{(i,j_{\mathrm{L}})}, \mathsf{crs}_{\mathsf{BC}}^{(i,j_{\mathrm{R}})}, \mathsf{crs}_{\mathsf{BC}}^{(i,j_{\mathrm{OUT}})}, c^{(i,j_{\mathrm{L}})}, c^{(i,j_{\mathrm{R}})}, c^{(i,j_{\mathrm{OUT}})}\right)$$
$$w^{(i,j)} = \left(w_{j_{\mathrm{L}}}^{(i)}, w_{j_{\mathrm{R}}}^{(i)}, w_{j_{\mathrm{OUT}}}^{(i)}, \sigma^{(i,j_{\mathrm{L}})}, \sigma^{(i,j_{\mathrm{R}})}, \sigma^{(i,j_{\mathrm{OUT}})}\right).$$

Now, the following conditions hold:

- By Eq. (3.6), $\mathsf{BC.Verify}\left(\mathsf{crs}_{\mathsf{BC}}^{(i,j_{\mathsf{pos}})}, c^{(i,j_{\mathsf{pos}})}, w_{j_{\mathsf{pos}}}^{(i)}, \sigma^{(i,j_{\mathsf{pos}})}\right) = 1$ for each $\mathsf{pos} \in \{\mathrm{L}, \mathrm{R}, \mathrm{OUT}\}$.
- Since the wire values $w_1^{(i)}, \ldots, w_s^{(i)}$ are associated with the wire values of $C(k, \mathbf{z}_i)$, it holds that $w_{j_{\mathrm{OUT}}}^{(i)} = \mathrm{NAND}(w_{j_{\mathrm{L}}}^{(i)}, w_{j_{\mathrm{R}}}^{(i)})$.

By construction of $\mathcal{R}$, this means $(x^{(i,j)}, w^{(i,j)}) \in \mathcal{R}$ for each $i \in I$ and $j \in [s_{\mathsf{int}}]$. By completeness of Π_{BARG},

$$\mathsf{BARG.Verify}(\mathsf{crs}_{\mathsf{BARG}}, C_{\mathcal{R}}, (x^{(i,j)})_{i \in I, j \in [s_{\mathsf{int}}]}, \pi_{\mathsf{BARG}}) = 1.$$

Since both checks pass, we conclude that $\mathsf{Verify}(\mathsf{crs}, I, \mathbf{r}_I, \pi)$ outputs 1, as required.

Security. We now state the main security theorems for Construction 5 and defer their formal proofs to Sect. 3.1 and the full version of this paper [7].

Theorem 7 (Somewhat Computational Binding). *Suppose $\kappa(\lambda, m) \leq m^{\delta} \cdot p(\lambda)$ for some constant $\delta < 1$ and a fixed polynomial $p(\cdot)$. Then, if Π_{BC} is statistically binding in binding mode and Π_{BARG} is adaptively sound, it follows that Construction 5 satisfies somewhat computational binding.*

Theorem 8 (Computational Hiding). *Suppose Π_{BC} satisfies mode indistinguishability and Π_{LRwPRF} is a leakage-resilient weak PRF. If $\ell(\lambda, m) \geq \ell_{\mathsf{BARG}}(\lambda, ms_{\mathsf{int}}, |C_{\mathcal{R}}|)$, Construction 5 satisfies computational hiding.*

Parameter Instantiations. We now describe a possible instantiation of the underlying building blocks in Construction 5 to obtain a NIZK argument from a sub-exponentially-secure somewhere-sound BARG. We summarize our main result in Corollary 1.

– We instantiate the one-time dual-mode bit commitment scheme Π_{BC} using Naor's bit commitment scheme [31] (Theorem 3) based on one-way functions. With this instantiation, the size of the circuit $C_{\mathcal{R}}$ in Construction 5 can be bounded by $O(\lambda^{c_1})$ for some constant $c_1 \in \mathbb{N}$.
– We instantiate the leakage-resilient weak PRF Π_{LRwPRF} with the scheme based on one-way functions [22,35] (Theorem 2). Let ℓ be the leakage parameter for the leakage-resilient weak PRF. With this instantiation, the keys have length

at most $\kappa = \ell \cdot O(\lambda^2)$ and the inputs have length $n = O(\lambda + \log \ell)$. Let s be the size of the Boolean circuit that takes as input a key k and an input $\mathbf{z}$ and outputs $\mathsf{LRwPRF.Eval}(k, \mathbf{z})$. Since the construction is efficient, the size of this circuit can be upper-bounded by $s = O((\ell\lambda)^{c_2})$ for some constant $c_2 \in \mathbb{N}$.

– We instantiate the batch argument Π_{BARG} with a scheme where the proof size satisfies

$$\ell_{\mathsf{BARG}}(\lambda, T, s) < T^{\varepsilon} \cdot O((\lambda + s)^{c_3}),$$

where $0 < \varepsilon < 1/(1 + c_2)$ and $c_3 \in \mathbb{N}$ are fixed constants. Existing BARG constructions based on standard number-theoretic assumptions [12,14,16,24,39] satisfy an even stronger succinctness guarantee where $\ell_{\mathsf{BARG}}(\lambda, T, s) \leq \mathsf{poly}(\lambda + \log T + s)$ where the proof size is *polylogarithmic* in the number of instances T; this is the default definition from Definition 6. However, as we show here, our construction applies even in settings where the BARG proof size scales with T^{ε} for sufficiently small constants $\varepsilon < 1$. Finally, all of the aforementioned BARG constructions satisfy somewhere soundness, which can be bootstrapped to an adaptively-sound construction using Theorem 20 (via complexity leveraging).

– With this choice of parameters, we choose constants $\delta_1 = \varepsilon/(1 - \varepsilon c_2)$ and $\delta_2 = (\varepsilon c_2 + c_1 c_3)/(1 - \varepsilon c_2)$. Finally, we set $\ell(\lambda, m) = m^{\delta_1} \cdot \Theta(\lambda^{\delta_2})$.

It is easy to see that this setting of parameters satisfies the requirements in Theorems 7 and 8:

– **Binding (Theorem 7):** For this choice of parameters,

$$\kappa(\lambda, m) = \ell(\lambda, m) \cdot O(\lambda^2) = m^{\delta_1} \cdot \mathsf{poly}(\lambda) = m^{\varepsilon/(1-\varepsilon c_2)} \cdot \mathsf{poly}(\lambda).$$

Moreover, since $\varepsilon < 1/(1 + c_2)$, this means $\varepsilon + \varepsilon c_2 < 1$ so $\varepsilon < 1 - \varepsilon c_2$. Correspondingly, this means that $0 < \varepsilon/(1 - \varepsilon c_2) < 1$. Thus, the condition of Theorem 7 is satisfied.

– **Hiding (Theorem 8):** For this choice of parameters, we have

$$\begin{aligned}
\ell_{\mathsf{BARG}}(\lambda, ms_{\mathsf{inp}}, |C_{\mathcal{R}}|) &< (ms)^{\varepsilon} \cdot O((\lambda + \lambda^{c_1})^{c_3}) \\
&= m^{\varepsilon}(\ell\lambda)^{\varepsilon c_2} \cdot O(\lambda^{c_1 c_3}) \\
&= m^{\varepsilon(1+\delta_1 c_2)} \cdot O\big(\lambda^{(\delta_2+1)(\varepsilon c_2)+c_1 c_3}\big).
\end{aligned}$$

Next, we can write

$$\begin{aligned}
\varepsilon(1 + \delta_1 c_2) &= \varepsilon\left(1 + \frac{\varepsilon c_2}{1 - \varepsilon c_2}\right) = \frac{\varepsilon}{1 - \varepsilon c_2} = \delta_1 \\
(\delta_2 + 1)\varepsilon c_2 + c_1 c_3 &= \delta_2 \varepsilon c_2 + \varepsilon c_2 + c_1 c_3 \\
&= (\varepsilon c_2 + c_1 c_3)\left(\frac{\varepsilon c_2}{1 - \varepsilon c_2} + 1\right) \\
&= \frac{\varepsilon c_2 + c_1 c_3}{1 - \varepsilon c_2} \\
&= \delta_2.
\end{aligned}$$

Correspondingly, we see that for this choice of parameters,

$$\begin{aligned}\ell_{\mathsf{BARG}}(\lambda, ms_{\mathsf{int}}, |C_{\mathcal{R}}|) &< m^{\varepsilon(1+\delta_1 c_2)} \cdot O\big(\lambda^{(\delta_2+1)(\varepsilon c_2)+c_1 c_3}\big) \\ &= m^{\delta_1} \cdot O(\lambda^{\delta_2}) \\ &= \ell(\lambda, m),\end{aligned}$$

which satisfies the requirement in Theorem 8.

We summarize this instantiation with the following corollary.

Corollary 1 (NIZKs from Sub-Exponentially Secure Somewhere-Sound BARGs). *Assuming the existence of one-way functions and either (1) an adaptively-sound BARG for* NP *(Definition 6);* or *(2) a sub-exponentially-secure somewhere-sound BARG for* NP *(Definition 7), there exists a computational NIZK argument for* NP.

3.1 Proof of Theorem 8 (Computational Hiding)

Let $\mathcal{A}$ be an efficient adversary for the computational hiding security game for Construction 5. We define a sequence of hybrid experiments between the challenger and $\mathcal{A}$.

- Hyb_0: This is the computational hiding game with $\beta = 0$. Specifically, the game proceeds as follows:
  - On input the security parameter 1^λ, algorithm $\mathcal{A}$ outputs the output length 1^m and a subset $I \subseteq [m]$.
  - The challenger samples $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^m)$. Specifically, it starts by sampling the common reference strings for the bit commitment schemes:
    - **CRS for the key:** For $j \in S_{\mathsf{key}}$, sample $\mathsf{crs}_{\mathsf{BC}}^{(\mathsf{key},j)} \leftarrow \mathsf{BC.Setup}\mathit{Bind}(1^\lambda)$.
    - **CRS for the evaluation point:** For $i \in [m]$ and $j \in S_{\mathsf{eval}}$, sample $\mathsf{crs}_{\mathsf{BC}}^{(i,j)} \leftarrow \mathsf{BC.SetupBind}(1^\lambda)$.
    - **CRS for the non-input wires:** For $i \in [m]$ and $j \in S_{\mathsf{int}}$, sample $\mathsf{crs}_{\mathsf{BC}}^{(i,j)} \leftarrow \mathsf{BC.SetupBind}(1^\lambda)$.

    Next, the challenger samples $\mathbf{z}_1, \ldots, \mathbf{z}_m \xleftarrow{\mathrm{R}} \{0,1\}^n$. For each each $i \in [m]$ and $j \in S_{\mathsf{eval}}$, it computes $(c^{(i,j)}, \sigma^{(i,j)}) \leftarrow \mathsf{BC.Commit}\big(\mathsf{crs}_{\mathsf{BC}}^{(i,j)}, z_{i,j-\kappa}\big)$. Finally, it samples $\mathsf{crs}_{\mathsf{BARG}} \leftarrow \mathsf{BARG.Setup}(1^\lambda, 1^{ms_{\mathsf{int}}}, 1^{|C_{\mathcal{R}}|})$ and defines crs according to Eq. (3.1).
  - Next, the challenger computes $(\mathbf{r}, \mathsf{st}) \leftarrow \mathsf{GenBits}(\mathsf{crs})$ and $\pi \leftarrow \mathsf{Prove}(\mathsf{st}, I)$. First, the challenger samples a weak PRF key $k \leftarrow \mathsf{LRwPRF.Setup}(1^\lambda, 1^\ell)$ and computes $r_i \leftarrow \mathsf{LRwPRF.Eval}(k, \mathbf{z}_i)$. It sets $\mathbf{r} = r_1 \| \cdots \| r_m$. To construct the proof π, the challenger first sets $w_1^{(i)}, \ldots, w_s^{(i)}$ to be the wire values of $C(k, \mathbf{z}_i)$ for each $i \in [m]$. Then, it does the following:
    - **Commit to the bits of k:** For each $j \in S_{\mathsf{key}}$, compute the commitment and opening $\big(c^{(\mathsf{key},j)}, \sigma^{(\mathsf{key},j)}\big) \leftarrow \mathsf{BC.Commit}\big(\mathsf{crs}^{(\mathsf{key},j)}, k_j\big)$.

* **Commit to the non-input wires for $C(k, \mathbf{z}_i)$:** For each $i \in [m]$ and $j \in S_{\mathsf{int}}$, compute $\left(c^{(i,j)}, \sigma^{(i,j)}\right) \leftarrow \mathsf{BC.Commit}\left(\mathsf{crs}^{(i,j)}, w_j^{(i)}\right)$.
* **Construct a BARG proof of validity:** For each $j \in [s_{\mathsf{int}}]$, let $j_{\mathrm{L}}, j_{\mathrm{R}}$ be the wire indices that determine the value of the j^{th} non-input wire $j_{\mathrm{OUT}} = (\kappa + n) + j$. Then, for each $i \in [m]$, define the statement $x^{(i,j)}$ and witness $w^{(i,j)}$ as follows:

$$x^{(i,j)} = \left(\mathsf{crs}_{\mathsf{BC}}^{(i,j_{\mathrm{L}})}, \mathsf{crs}_{\mathsf{BC}}^{(i,j_{\mathrm{R}})}, \mathsf{crs}_{\mathsf{BC}}^{(i,j_{\mathrm{OUT}})}, c^{(i,j_{\mathrm{L}})}, c^{(i,j_{\mathrm{R}})}, c^{(i,j_{\mathrm{OUT}})}\right)$$
$$w^{(i,j)} = \left(w_{j_{\mathrm{L}}}^{(i)}, w_{j_{\mathrm{R}}}^{(i)}, w_{j_{\mathrm{OUT}}}^{(i)}, \sigma^{(i,j_{\mathrm{L}})}, \sigma^{(i,j_{\mathrm{R}})}, \sigma^{(i,j_{\mathrm{OUT}})}\right).$$

As in Construction 5, we adopt the convention that $\mathsf{crs}_{\mathsf{BC}}^{(i,j)} := \mathsf{crs}_{\mathsf{BC}}^{(\mathsf{key},j)}$, $c^{(i,j)} := c^{(\mathsf{key},j)}$, and $\sigma^{(i,j)} := \sigma^{(\mathsf{key},j)}$. The challenger computes π_{BARG} according to Eq. (3.4).

Finally the challenger defines the proof π to be

$$\pi = \left(\pi_{\mathsf{BARG}}, \{c^{(\mathsf{key},j)}\}_{j \in S_{\mathsf{key}}}, \{c^{(i,j)}\}_{i \in I, j \in S_{\mathsf{int}}}, \{\sigma^{(i,s)}\}_{i \in I}\right).$$

- The challenger gives $(\mathsf{crs}, I, \mathbf{r}_I, \pi, \mathbf{r}_{\overline{I}})$ to $\mathcal{A}$. Algorithm $\mathcal{A}$ outputs a bit $b' \in \{0,1\}$, which is the output of the experiment.

– Hyb_1: Same as Hyb_0, except the challenger switches the bit commitments for the key and the non-input wires to be equivocating.

- On input the security parameter 1^λ, algorithm $\mathcal{A}$ outputs the output length 1^m and a subset $I \subseteq [m]$.
- The challenger starts by sampling the common reference strings for the bit commitment schemes:
  * **CRS for the key:** For $j \in S_{\mathsf{key}}$, sample

$$\left(\mathsf{crs}_{\mathsf{BC}}^{(\mathsf{key},j)}, \tilde{c}^{(\mathsf{key},j)}, \tilde{\sigma}_0^{(\mathsf{key},j)}, \tilde{\sigma}_1^{(\mathsf{key},j)}\right) \leftarrow \mathsf{BC.SetupEquivocate}(1^\lambda).$$

  * **CRS for the evaluation point:** For $i \in [m]$ and $j \in S_{\mathsf{eval}}$, sample $\mathsf{crs}_{\mathsf{BC}}^{(i,j)} \leftarrow \mathsf{BC.SetupBind}(1^\lambda)$.
  * **CRS for the non-input wires:** For $i \in [m]$ and $j \in S_{\mathsf{int}}$, sample

$$\left(\mathsf{crs}_{\mathsf{BC}}^{(i,j)}, \tilde{c}^{(i,j)}, \tilde{\sigma}_0^{(i,j)}, \tilde{\sigma}_1^{(i,j)}\right) \leftarrow \mathsf{BC.SetupEquivocate}(1^\lambda).$$

  The remaining components of the crs are computed exactly as described in Hyb_0 (same for all hybrids).
- The challenger samples $k \leftarrow \mathsf{LRwPRF.Setup}(1^\lambda, 1^\ell)$ and computes $r_i \leftarrow \mathsf{LRwPRF.Eval}(k, \mathbf{z}_i)$. It sets $\mathbf{r} = r_1 \| \cdots \| r_m$. To construct the proof π, the challenger first sets $w_1^{(i)}, \ldots, w_s^{(i)}$ to be the wire values of $C(k, \mathbf{z}_i)$ for each $i \in [m]$. Then, it does the following:
  * **Commit to the bits of k:** For each $j \in S_{\mathsf{key}}$, let $c^{(\mathsf{key},j)} := \tilde{c}^{(\mathsf{key},j)}$ and $\sigma^{(\mathsf{key},j)} := \tilde{\sigma}_b^{(\mathsf{key},j)}$ where $b = k_j$.

* **Commit to the non-input wires for $C(k, \mathbf{z}_i)$:** For each $i \in [m]$ and $j \in S_{\mathsf{int}}$, let $c^{(i,j)} := \tilde{c}^{(i,j)}$ and $\sigma^{(i,j)} := \tilde{\sigma}_b^{(i,j)}$ where $b = w_j^{(i)}$.
* **Construct a BARG proof of validity:** The BARG proof π_{BARG} is computed exactly as described in Hyb_0 (same for all hybrids).

The challenger defines the proof π as in Hyb_0 (same for all hybrids).

- The challenger gives $(\mathsf{crs}, I, \mathbf{r}_I, \pi, \mathbf{r}_{\overline{I}})$ to $\mathcal{A}$. Algorithm $\mathcal{A}$ outputs a bit $b' \in \{0,1\}$, which is the output of the experiment.

– Hyb_2: Same as Hyb_1, except for all $i \notin I$, the challenger samples $r_i \xleftarrow{R} \{0,1\}$. Specifically, the experiment proceeds as follows:

- On input the security parameter 1^λ, algorithm $\mathcal{A}$ outputs the output length 1^m and a subset $I \subseteq [m]$.
- The challenger starts by sampling the common reference strings for the bit commitment schemes:
  * **CRS for the key:** For $j \in S_{\mathsf{key}}$, sample
  $$\left(\mathsf{crs}_{\mathsf{BC}}^{(\mathsf{key},j)}, \tilde{c}^{(\mathsf{key},j)}, \sigma_0^{(\mathsf{key},j)}, \sigma_1^{(\mathsf{key},j)}\right) \leftarrow \mathsf{BC.SetupEquivocate}(1^\lambda).$$
  * **CRS for the evaluation point:** For $i \in [m]$ and $j \in S_{\mathsf{eval}}$, sample $\mathsf{crs}_{\mathsf{BC}}^{(i,j)} \leftarrow \mathsf{BC.SetupBind}(1^\lambda)$.
  * **CRS for the non-input wires:** For $i \in [m]$ and $j \in S_{\mathsf{int}}$, sample
  $$\left(\mathsf{crs}_{\mathsf{BC}}^{(i,j)}, \tilde{c}^{(i,j)}, \tilde{\sigma}_0^{(i,j)}, \tilde{\sigma}_1^{(i,j)}\right) \leftarrow \mathsf{BC.SetupEquivocate}(1^\lambda).$$

  The remaining components of the crs are computed exactly as described in Hyb_0 (same for all hybrids).
- The challenger samples $k \leftarrow \mathsf{LRwPRF.Setup}(1^\lambda, 1^\ell)$. For each $i \in I$, it computes $r_i \leftarrow \mathsf{LRwPRF.Eval}(k, \mathbf{z}_i)$. For each $i \notin I$, it samples $r_i \xleftarrow{R} \{0,1\}$. It sets $\mathbf{r} = r_1 \| \cdots \| r_m$. To construct the proof π, the challenger first sets $w_1^{(i)}, \ldots, w_s^{(i)}$ to be the wire values of $C(k, \mathbf{z}_i)$ for each $i \in [m]$. Then, it does the following:
  * **Commit to the bits of k:** For each $j \in S_{\mathsf{key}}$, let $c^{(\mathsf{key},j)} := \tilde{c}^{(\mathsf{key},j)}$ and $\sigma^{(\mathsf{key},j)} := \tilde{\sigma}_b^{(\mathsf{key},j)}$ where $b = k_j$.
  * **Commit to the non-input wires for $C(k, \mathbf{z}_i)$:** For each $i \in [m]$ and $j \in S_{\mathsf{int}}$, let $c^{(i,j)} := \tilde{c}^{(i,j)}$ and $\sigma^{(i,j)} := \tilde{\sigma}_b^{(i,j)}$ where $b = w_j^{(i)}$.
  * **Construct a BARG proof of validity:** The BARG proof π_{BARG} is computed exactly as described in Hyb_0 (same for all hybrids).

  The challenger defines the proof π as in Hyb_0 (same for all hybrids).
- The challenger gives $(\mathsf{crs}, I, \mathbf{r}_I, \pi, \mathbf{r}_{\overline{I}})$ to $\mathcal{A}$. Algorithm $\mathcal{A}$ outputs a bit $b' \in \{0,1\}$, which is the output of the experiment.

– Hyb_3: Same as Hyb_2 except the challenger switches the commitments back to binding mode. This is the computational hiding game with $\beta = 1$. Specifically, the game proceeds as follows:

- On input the security parameter 1^λ, algorithm $\mathcal{A}$ outputs the output length 1^m and a subset $I \subseteq [m]$.
- The challenger starts by sampling the common reference strings for the bit commitment schemes:

  - ∗ **CRS for the key:** For $j \in S_{\mathsf{key}}$, sample $\mathsf{crs}_{\mathsf{BC}}^{(\mathsf{key},j)} \leftarrow \mathsf{BC.SetupBind}(1^\lambda)$.
  - ∗ **CRS for the evaluation point:** For $i \in [m]$ and $j \in S_{\mathsf{eval}}$, sample $\mathsf{crs}_{\mathsf{BC}}^{(i,j)} \leftarrow \mathsf{BC.SetupBind}(1^\lambda)$.
  - ∗ **CRS for the non-input wires:** For $i \in [m]$ and $j \in S_{\mathsf{int}}$, sample $\mathsf{crs}_{\mathsf{BC}}^{(i,j)} \leftarrow \mathsf{BC.SetupBind}(1^\lambda)$.

  The remaining components of the crs are computed exactly as described in Hyb_0 (same for all hybrids).
- The challenger samples $k \leftarrow \mathsf{LRwPRF.Setup}(1^\lambda, 1^\ell)$. For each $i \in I$, it computes $r_i \leftarrow \mathsf{LRwPRF.Eval}(k, \mathbf{z}_i)$. For each $i \notin I$, it samples $r_i \xleftarrow{\mathrm{R}} \{0,1\}$. It sets $\mathbf{r} = r_1 \| \cdots \| r_m$. To construct the proof π, the challenger first sets $w_1^{(i)}, \ldots, w_s^{(i)}$ to be the wire values of $C(k, \mathbf{z}_i)$ for each $i \in [m]$. Then, it does the following:
  - ∗ **Commit to the bits of k:** For each $j \in S_{\mathsf{key}}$, compute the commitment and opening $\left(c^{(\mathsf{key},j)}, \sigma^{(\mathsf{key},j)}\right) \leftarrow \mathsf{BC.Commit}\left(\mathsf{crs}^{(\mathsf{key},j)}, k_j\right)$.
  - ∗ **Commit to the non-input wires for $C(k, \mathbf{z}_i)$:** For each $i \in [m]$ and $j \in S_{\mathsf{int}}$, compute $\left(c^{(i,j)}, \sigma^{(i,j)}\right) \leftarrow \mathsf{BC.Commit}\left(\mathsf{crs}^{(i,j)}, w_j^{(i)}\right)$.
  - ∗ **Construct a BARG proof of validity:** The BARG proof π_{BARG} is computed exactly as described in Hyb_0 (same for all hybrids).

  The challenger defines the proof π as in Hyb_0 (same for all hybrids).
- The challenger gives $(\mathsf{crs}, I, \mathbf{r}_I, \pi, \mathbf{r}_{\overline{I}})$ to $\mathcal{A}$. Algorithm $\mathcal{A}$ outputs a bit $b' \in \{0,1\}$, which is the output of the experiment.

We write $\mathsf{Hyb}_i(\mathcal{A})$ to denote the output distribution of an execution of Hyb_i with adversary $\mathcal{A}$. Due to space limitations, we give the formal analysis in the full version of this paper [7].

4 Hidden-Bits Generator from BARGs and Public-Key Encryption

In this section, we show how to construct a hidden-bits generator with subset-dependent proofs by combining a *polynomial-hard* somewhere-sound BARG with a public-key encryption scheme. Compared to Corollary 1, this construction only relies on polynomial hardness on the somewhere-sound BARG (as opposed to sub-exponential hardness), but in exchange, it requires an additional assumption of public-key encryption. As described in Sect. 1.1, this construction follows the same template as the previous construction (Sect. 3), but uses public-key encryption to construct a one-time dual-mode bit commitment with *efficient* extraction.

4.1 One-Time Dual-Mode Bit Commitment with Extraction

The main building block we use in this section is a one-time dual-mode bit commitment scheme that supports *efficient* extraction. Recall that in a standard

one-time dual-mode bit commitment scheme (Definition 3), we only require the bit commitment scheme to be statistically binding in binding mode. Here, we upgrade the statistical binding to a strong extractability guarantee. This will allow us to base security of our hidden-bits generator somewhere soundness rather than adaptive soundness.

Definition 9 (One-Time Dual-Mode Bit Commitment with Extraction). *A one-time dual-mode bit commitment with extraction is a tuple of algorithms* $\Pi_{\mathsf{BC}} = (\mathsf{SetupBind}, \mathsf{SetupEquivocate}, \mathsf{Commit}, \mathsf{Verify}, \mathsf{Extract})$ *with the following syntax:*

- $\mathsf{SetupBind}(1^\lambda) \to (\mathsf{crs}, \mathsf{td})$*: On input the security parameter* λ*, the setup algorithm for the binding mode outputs a common reference string* crs *and trapdoor* td*.*
- $\mathsf{SetupEquivocate}(1^\lambda) \to (\mathsf{crs}, \tilde{c}, \tilde{\sigma}_0, \tilde{\sigma}_1)$*: On input the security parameter* λ*, the setup algorithm for the equivocating mode outputs a common reference string* crs *along with a commitment* $\tilde{c}$ *and openings* $\tilde{\sigma}_0, \tilde{\sigma}_1$*.*
- $\mathsf{Commit}(\mathsf{crs}, b) \to (c, \sigma)$*: On input the common reference string* crs *and a bit* $b \in \{0,1\}$*, the commit algorithm outputs a commitment* c *and an opening* σ*.*
- $\mathsf{Verify}(\mathsf{crs}, c, b, \sigma) \to \{0,1\}$*: On input the common reference string* crs*, a commitment* c*, a bit* $b \in \{0,1\}$*, and an opening* σ*, the verification algorithm outputs a bit* $b' \in \{0,1\}$*.*
- $\mathsf{Extract}(\mathsf{td}, c) \to \{0,1\}$*: On input the trapdoor* td *and a commitment* c*, the verification algorithm outputs a bit* $b \in \{0,1\}$*.*

We require Π_{BC} *satisfy the following properties:*

- **Correctness:** *Same as in Definition 3.*
- **Mode indistinguishability:** *Same as in Definition 3.*
- **Extractable in binding mode:** *For all adversaries* $\mathcal{A}$*, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that for all* $\lambda \in \mathbb{N}$*,*

$$\Pr\left[\mathsf{Verify}(\mathsf{crs}, c, b, \sigma) = 1 \wedge b \neq b' : \begin{array}{c}(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{SetupBind}(1^\lambda) \\ (c, \sigma, b) \leftarrow \mathcal{A}(1^\lambda, \mathsf{crs}) \\ b' \leftarrow \mathsf{Extract}(\mathsf{td}, c)\end{array}\right] = \mathsf{negl}(\lambda).$$

Constructing a One-Time Dual-Mode Bit Commitment with Extraction Scheme. We can construct a one-time dual-mode bit commitment with extraction scheme by composing a vanilla one-time dual-mode bit commitment scheme (Definition 3) with a public-key encryption scheme. A similar approach was used implicitly in previous works [27,29].

Construction 9 (One-Time Dual-Mode Bit Commitment with Extraction). Our construction relies on the following primitives:

- Let $\Pi'_{\mathsf{BC}} = (\mathsf{BC.SetupBind}', \mathsf{BC.SetupEquivocate}', \mathsf{BC.Commit}', \mathsf{BC.Verify}')$ be a one-time dual mode bit commitment scheme. Let $\ell_{\mathsf{BC}} = \ell_{\mathsf{BC}}(\lambda)$ be a bound on the length of the openings output by $\mathsf{BC.Commit}'$.

– Let $\Pi_{\mathsf{PKE}} = (\mathsf{PKE.Setup}, \mathsf{PKE.Encrypt}, \mathsf{PKE.Decrypt})$ be a public-key encryption scheme with message space $\mathcal{M}_\lambda = \{0,1\}^{\ell_{\mathsf{BC}}(\lambda)+1} \cup \{\bot\}$. Let $\rho = \rho(\lambda)$ be the randomness complexity of $\mathsf{PKE.Encrypt}$ (i.e., the number of bits of randomness that $\mathsf{PKE.Encrypt}$ takes as input).

We construct a one-time dual mode bit commitment scheme scheme with extraction $\Pi_{\mathsf{BC}} = (\mathsf{SetupBind}, \mathsf{SetupEquivocate}, \mathsf{Commit}, \mathsf{Verify}, \mathsf{Extract})$ as follows:

– $\mathsf{SetupBind}(1^\lambda)$: On input the security parameter λ, the binding mode setup algorithm samples $\mathsf{crs}' \leftarrow \mathsf{BC.SetupBind}'(1^\lambda)$ and $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{PKE.Setup}(1^\lambda)$. It then outputs the common reference string $\mathsf{crs} = (\mathsf{crs}', \mathsf{pk})$ and the extraction trapdoor $\mathsf{td} = (\mathsf{crs}', \mathsf{sk})$.
– $\mathsf{SetupEquivocate}(1^\lambda)$: On input the security parameter λ, the equivocating mode setup algorithm samples $(\mathsf{crs}', \tilde{c}', \tilde{\sigma}'_0, \tilde{\sigma}'_1) \leftarrow \mathsf{BC.SetupEquivocate}'(1^\lambda)$ and $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{PKE.Setup}(1^\lambda)$. Next, for $b \in \{0,1\}$, it samples $r_b \xleftarrow{R} \{0,1\}^\rho$ and sets $\mathsf{ct}_b \leftarrow \mathsf{PKE.Encrypt}(\mathsf{pk}, (b, \tilde{\sigma}'_b); r_b)$. It outputs the common reference string $\mathsf{crs} = (\mathsf{crs}', \mathsf{pk})$, the commitment $\tilde{c} = (\tilde{c}', \mathsf{ct}_0, \mathsf{ct}_1)$, and the openings $\tilde{\sigma}_0 = (\tilde{\sigma}'_0, r_0)$, $\tilde{\sigma}_1 = (\tilde{\sigma}'_1, r_1)$.
– $\mathsf{Commit}(\mathsf{crs}, b)$: On input the common reference string $\mathsf{crs} = (\mathsf{crs}', \mathsf{pk})$ and a bit $b \in \{0,1\}$, the commit algorithm constructs a commitment $(c', \sigma') \leftarrow \mathsf{BC.Commit}'(\mathsf{crs}', b)$. Then, it samples $r_b \xleftarrow{R} \{0,1\}^\rho$ and computes $\mathsf{ct}_b \leftarrow \mathsf{PKE.Encrypt}(\mathsf{pk}, (b, \sigma'); r_b)$. It also computes $\mathsf{ct}_{1-b} \leftarrow \mathsf{PKE.Encrypt}(\mathsf{pk}, \bot)$. Finally, it outputs the commitment $c = (c', \mathsf{ct}_0, \mathsf{ct}_1)$ and the opening $\sigma = (\sigma', r_b)$.
– $\mathsf{Verify}(\mathsf{crs}, c, b, \sigma)$: On input the common reference string $\mathsf{crs} = (\mathsf{crs}', \mathsf{pk})$, a commitment $c = (c', \mathsf{ct}_0, \mathsf{ct}_1)$, a bit $b \in \{0,1\}$, and an opening $\sigma = (\sigma', r_b)$, the verification algorithm outputs 1 if $\mathsf{BC.Verify}'(\mathsf{crs}', c', b, \sigma') = 1$ and $\mathsf{ct}_b = \mathsf{PKE.Encrypt}(\mathsf{pk}, (b, \sigma'); r_b)$.
– $\mathsf{Extract}(\mathsf{td}, c)$: On input an extraction trapdoor $\mathsf{td} = (\mathsf{crs}', \mathsf{sk})$ and a commitment $c = (c', \mathsf{ct}_0, \mathsf{ct}_1)$, the extraction algorithm computes the message $m_0 \leftarrow \mathsf{PKE.Decrypt}(\mathsf{sk}, \mathsf{ct}_0)$. If $m_0 = (0, \sigma')$ and $\mathsf{BC.Verify}'(\mathsf{crs}', c', 0, \sigma') = 1$, then it outputs 0. Otherwise, it outputs 1.

Correctness and Security Analysis. We state correctness and security theorems here, but defer the proofs to the full version of this paper [7].

Theorem 10 (Correctness). *If Π'_{BC} and Π_{PKE} are correct, then Construction 9 is correct.*

Theorem 11 (Mode Indistinguishability). *Suppose Π'_{BC} satisfies mode indistinguishability. Then Construction 9 satisfies mode indistinguishability.*

Theorem 12 (Extractable in Binding Mode). *If Π_{PKE} is perfectly correct and Π'_{BC} is statistically binding, then Construction 9 is extractable in binding mode.*

Corollary 2 (One-Time Dual-Mode Bit Commitment with Extraction). *Assuming the existence of public-key encryption, there exists a one-time dual-mode bit commitment with extraction scheme.*

4.2 Hidden-Bits Generator Construction

We now describe our hidden-bits generator based on public-key encryption. The construction replaces the one-time dual-mode bit commitment scheme in Construction 5 with a scheme that supports extraction. This allows basing security on *polynomial* somewhere soundness of the underlying BARG rather than adaptive soundness (which necessitated sub-exponential hardness). The construction is identical to Construction 5, except we modify the binding analysis to rely on somewhere soundness of the BARG (and extraction) rather than adaptive soundness.

Construction 13 (Hidden-Bits Generator from Polynomial-Hard Batch Arguments). The construction is identical to Construction 5, except for the following two differences:

- We replace the one-time dual-mode bit commitment scheme with a one-time dual-mode bit commitment scheme with extraction $\Pi_{\mathsf{BC}} = (\mathsf{SetupBind}, \mathsf{SetupEquivocate}, \mathsf{Commit}, \mathsf{Verify}, \mathsf{Extract})$. Note that the $\mathsf{Extract}$ algorithm is only needed in the security analysis, so the scheme semantics are identical to Construction 5.
- We replace the adaptively-sound BARG with a somewhere sound BARG. Functionally-speaking, the only difference in Setup is when sampling the CRS for the BARG, the scheme additionally provides a dummy index 1. Namely, the Setup algorithm samples $\mathsf{crs}_{\mathsf{BARG}} \leftarrow \mathsf{BARG.Setup}(1^\lambda, 1^{ms_{\mathsf{int}}}, 1^{|C_{\mathcal{R}}|}, 1)$.

Correctness and Security Analysis. We now state the correctness and security theorems for Construction 13. The correctness and hiding proofs are identical to the respective proofs for Construction 5 (Theorem 6 and Theorem 8). We defer the binding proof to the full version of this paper [7].

Theorem 14 (Correctness). *If Π_{BARG} is complete and Π_{BC} is correct, then Construction 13 is correct.*

Proof. Follows by the same argument as in the proof of Theorem 6.

Theorem 15 (Somewhat Computational Binding). *Suppose $\kappa(\lambda, m) \leq m^\delta \cdot p(\lambda)$ for some constant $\delta < 1$ and a fixed polynomial $p(\cdot)$. If Π_{BC} is extractable in binding mode and Π_{BARG} is somewhere sound, then Construction 13 satisfies somewhat computational binding.*

Theorem 16 (Computational Hiding). *Suppose Π_{BC} satisfies mode indistinguishability and Π_{LRwPRF} is a secure leakage-resilient weak PRF. If $\ell(\lambda, m) \geq \ell_{\mathsf{BARG}}(\lambda, ms_{\mathsf{int}}, |C_{\mathcal{R}}|)$, Construction 13 satisfies computational hiding.*

Proof. Follows by the same argument as in the proof of Theorem 8 (see Sect. 3.1).

Parameter Instantiations. We can instantiate the underlying primitives following the same methodology as in Sect. 3. This yields the following corollary:

Corollary 3 (NIZKs from Somewhere-Sound BARGs and PKE). *Assuming the existence of public-key encryption and somewhere-sound BARGs for* NP *(Definitions 6 and 7), there exists a computational NIZK argument for* NP*.*

5 BARGs with Adaptive Soundness via Sub-Exponential Hardness

In this section, we show that using complexity leveraging, any sub-exponentially-secure somewhere-sound BARG (Definition 7) is also adaptively sound (Definition 6). Specifically, the analysis relies on sub-exponential index hiding. We describe our construction and analysis below:

Construction 17 (Adaptively-Sound BARG from a Somewhere-Sound BARG). Let λ be a security parameter and s be a circuit-size parameter. Let $\Pi_{\mathsf{SSBARG}} = (\mathsf{SSBARG.Setup}, \mathsf{SSBARG.Prove}, \mathsf{SSBARG.Verify})$ be a somewhere-sound batch argument for Boolean circuit satisfiability. Let $\lambda_{\mathsf{SSBARG}} = \lambda_{\mathsf{SSBARG}}(\lambda, s)$ be a polynomial which will be set in the security proof (Theorem 20). We construct an adaptively-secure batch argument for Boolean circuit satisfiability $\Pi_{\mathsf{BARG}} = (\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify})$ as follows:

- $\mathsf{Setup}(1^\lambda, 1^T, 1^s)$: Output

$$\mathsf{crs} \leftarrow \mathsf{SSBARG.Setup}(1^{\lambda_{\mathsf{SSBARG}}(\lambda,s)}, 1^T, 1^s, 1).$$

- $\mathsf{Prove}(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_t), (\mathbf{w}_1, \ldots, \mathbf{w}_t))$: Output

$$\pi \leftarrow \mathsf{SSBARG.Prove}(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_t), (\mathbf{w}_1, \ldots, \mathbf{w}_t)).$$

- $\mathsf{Verify}(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_t), \pi)$: Output

$$b \leftarrow \mathsf{SSBARG.Verify}(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_t), \pi).$$

Correctness and Security Analysis. We state the correctness and security theorems for Construction 17 here, but defer the proof of soundness to the full version of this paper [7].

Theorem 18 (Completeness). *If Π_{SSBARG} is complete, then Construction 17 is complete.*

Proof. Follows immediately from completeness of Π_{SSBARG}.

Theorem 19 (Succinctness). *If Π_{SSBARG} is succinct, then Construction 17 is succinct.*

Proof. Take any common reference string crs in the support of $\mathsf{Setup}(1^\lambda, 1^T, 1^s)$. Then, crs is in the support of $\mathsf{SSBARG.Setup}(1^{\lambda_{\mathsf{SSBARG}}}, 1^T, 1^s, 1)$. Let $C\colon \{0,1\}^n \times \{0,1\}^h \to \{0,1\}$ be a Boolean circuit of size at most s. Take any sequence of statements $\mathbf{x}_1, \ldots, \mathbf{x}_t \in \{0,1\}^n$ and witnesses $\mathbf{w}_1, \ldots, \mathbf{w}_t \in \{0,1\}^h$, where $t \leq T$. Let proof π be $\pi \leftarrow \mathsf{Prove}(\mathsf{crs}, C, (\mathbf{x}_1, \ldots, \mathbf{x}_t), (\mathbf{w}_1, \ldots, \mathbf{w}_t))$. By succinctness of Π_{SSBARG}, $|\pi| \leq p(\lambda_{\mathsf{SSBARG}} + \log t + s)$ for some fixed polynomial $p(\cdot)$. Next, $\lambda_{\mathsf{SSBARG}} = (\lambda + s)^c$ for some constant $c \in \mathbb{N}$. Thus, we conclude that $|\pi| \leq p((\lambda + s)^c + \log t + s) \leq q(\lambda + \log t + s)$, for a fixed polynomial q that depends only on the polynomial p and the constant c.

Theorem 20 (Adaptive Soundness). *Suppose Π_{SSBARG} is a somewhere-sound BARG which satisfies sub-exponential index hiding with parameter $c > 1$ and somewhere soundness. Suppose moreover that $\lambda_{\mathsf{SSBARG}}(\lambda, s) = (\lambda + s)^c$. Then, Construction 17 is adaptively sound.*

Acknowledgments. Brent Waters is supported by NSF CNS-1908611, CNS-2318701, and a Simons Investigator award. David J. Wu is supported by NSF CNS-2151131, CNS-2140975, CNS-2318701, a Microsoft Research Faculty Fellowship, and a Google Research Scholar award.

References

1. Bellare, M., Hofheinz, D., Yilek, S.: Possibility and impossibility results for encryption and commitment secure under selective opening. In: EUROCRYPT (2009)
2. Bellare, M., Yung, M.: Certifying cryptographic tools: the case of trapdoor permutations. In: CRYPTO (1992)
3. Bitansky, N., Kamath, C., Paneth, O., Rothblum, R.D., Vasudevan, P.N.: Batch proofs are statistically hiding. IACR Cryptol. ePrint Arch. (2023). https://eprint.iacr.org/archive/2023/754/20230626:185215
4. Bitansky, N., Kamath, C., Paneth, O., Rothblum, R.D., Vasudevan, P.N.: Batch proofs are statistically hiding. IACR Cryptol. ePrint Arch. (2023). https://eprint.iacr.org/archive/2023/754/20231204:075616
5. Bitansky, N., Kamath, C., Paneth, O., Rothblum, R.D., Vasudevan, P.N.: Batch proofs are statistically hiding. IACR Cryptol. ePrint Arch. (2023). https://eprint.iacr.org/archive/2023/754/20230525:044715
6. Blum, M., Feldman, P. and Micali, S.: Non-interactive zero-knowledge and its applications (extended abstract). In: STOC (1988)
7. Bradley, E., Waters, B., Wu, D.J.: Batch arguments to NIZKs from one-way functions. Cryptology ePrint Archive, Paper 2023/1938 (2023). https://eprint.iacr.org/2023/1938
8. Brakerski, Z., Brodsky, M.F., Kalai, Y.T., Lombardi, A., Paneth, O.: SNARGs for monotone policy batch NP. In: CRYPTO (2023)
9. Canetti, R., Halevi, S., Katz, J.: A forward-secure public-key encryption scheme. In: EUROCRYPT (2003)
10. Canetti, R., Lichtenberg, A.: Certifying trapdoor permutations, revisited. In: TCC (2018)
11. Champion, J., Wu, D.J.: Non-interactive zero-knowledge from non-interactive batch arguments. In: CRYPTO (2023)

12. Choudhuri, A.R., Garg, S., Jain, A., Jin, Z., Zhang, J.: Correlation intractability and SNARGs from sub-exponential DDH. In: CRYPTO (2023)
13. Choudhuri, A.R., Jain, A., Jin, Z.: Non-interactive batch arguments for NP from standard assumptions. In: CRYPTO (2021)
14. Choudhuri, A.R., Jain, A., Jin, Z.: SNARGs for P from LWE. In: FOCS (2021)
15. Damgård, I., Nielsen, J.B.: Perfect hiding and perfect binding universally composable commitment schemes with constant expansion factor. In: CRYPTO (2002)
16. Devadas, L., Goyal, R., Kalai, Y., Vaikuntanathan, V.: Rate-1 non-interactive arguments for batch-NP and applications. In: FOCS (2022)
17. Feige, U., Lapidot, D., Shamir, A.: Multiple non-interactive zero knowledge proofs based on a single random string (extended abstract). In: FOCS (1990)
18. Garg, R., Sheridan, K., Waters, B., Wu, D.J.: Fully succinct batch arguments for NP from indistinguishability obfuscation. In: TCC (2022)
19. Gentry, C., Wichs, D.: Separating succinct non-interactive arguments from all falsifiable assumptions. In: STOC (2011)
20. Goldreich, O., Rothblum, R.D.: Enhancements of trapdoor permutations. J. Cryptol. **26**(3), 484–512 (2013)
21. Goldwasser, S., Micali, S., Rackoff, C.: The knowledge complexity of interactive proof-systems (extended abstract). In: STOC (1985)
22. Wee, H.C.L.A.A., Nguyen, H.W.D.J.T.: Leakage-resilient cryptography from minimal assumptions. In: EUROCRYPT (2013)
23. Hulett, J., Jawale, R., Khurana, D., Srinivasan, A.: SNARGs for P from sub-exponential DDH and QR. In: EUROCRYPT (2022)
24. Kalai, Y., Lombardi, A., Vaikuntanathan, V., Wichs, D.: Boosting batch arguments and RAM delegation. In: STOC (2023)
25. Kalai, Y.T., Vaikuntanathan, V., Zhang, R.Y.: Somewhere statistical soundness, post-quantum security, and SNARGs. In: TCC (2021)
26. Kitagawa, F., Matsuda, T., Yamakawa, T.: NIZK from SNARG. In: TCC (2020)
27. Koppula, V., Waters, B.: Realizing chosen ciphertext security generically in attribute-based encryption and predicate encryption. In: CRYPTO (2019)
28. Libert, B., Passelègue, A., Wee, H., Wu, D.J., New constructions of statistical NIZKs: Dual-mode DV-NIZKs and more. In: EUROCRYPT (2020)
29. Lombardi, A., Quach, W., Rothblum, R.D., Wichs, D., Wu, D.J.: New constructions of reusable designated-verifier NIZKs. In: CRYPTO (2019)
30. Matsuda, T.: Chosen ciphertext security via BARGs. IACR Cryptol. ePrint Arch. (2023)
31. Naor, M.: Bit commitment using pseudo-randomness. In: CRYPTO (1989)
32. Naor, M., Yung, M.: Public-key cryptosystems provably secure against chosen ciphertext attacks. In: STOC (1990)
33. Nassar, S., Waters, B., Wu, D.J.: Monotone policy BARGs from BARGs and additively homomorphic encryption. In: TCC (2024)
34. Quach, W., Rothblum, R.D., Wichs, D.: Reusable designated-verifier NIZKs for all NP from CDH. In: EUROCRYPT (2019)
35. Quach, W., Waters, B., Wichs, D.: Targeted lossy functions and applications. In: CRYPTO (2021)
36. De Santis, A., Di Crescenzo, G., Ostrovsky, R., Persiano, G., Sahai, A.: Robust non-interactive zero knowledge. In: CRYPTO, Rafail Ostrovsky (2001)
37. Waters, B.: A new approach for non-interactive zero-knowledge from learning with errors. In: STOC, pp. 399–410 (2024)

38. Waters, B., Wee, H., Wu, D.J.: New techniques for preimage sampling: improved NIZKs and more from LWE. Cryptology ePrint Archive, Paper 2023/1938 (2024). https://eprint.iacr.org/2023/1938
39. Waters, B., Wu, D.J.: Batch arguments for NP and more from standard bilinear group assumptions. In: CRYPTO (2022)
40. Waters, B., Wu, D.J.: Adaptively-sound succinct arguments for NP from indistinguishability obfuscation. In: STOC, pp. 387–398 (2024)
41. Waters, B., Wu, D.J.: A pure indistinguishability obfuscation approach to adaptively-sound SNARGs for NP. IACR Cryptol. ePrint Arch., p. 933 (2024)
42. Waters, B., Zhandry, M.: Adaptive security in SNARGs via iO and lossy functions. In: CRYPTO, pp. 72–104 (2024)

Security Bounds for Proof-Carrying Data from Straightline Extractors

Alessandro Chiesa[1], Ziyi Guan[1(✉)], Shahar Samocha[2], and Eylon Yogev[3]

[1] EPFL, Lausanne, Switzerland
{alessandro.chiesa,ziyi.guan}@epfl.ch
[2] StarkWare, Netanya, Israel
shahars@starkware.co
[3] Bar-Ilan University, Ramat Gan, Israel
eylon.yogev@biu.ac.il

Abstract. Proof-carrying data (PCD) is a powerful cryptographic primitive that allows mutually distrustful parties to perform distributed computation in an efficiently verifiable manner. Real-world deployments of PCD have sparked keen interest within the applied community and industry.

Known constructions of PCD are obtained by recursively-composing SNARKs or related primitives. Unfortunately, known security analyses incur expensive blowups, which practitioners have disregarded as the analyses would lead to setting parameters that are prohibitively expensive.

In this work we study the concrete security of recursive composition, with the goal of better understanding how to reasonably set parameters for certain PCD constructions of practical interest. Our main result is that PCD obtained from SNARKs with *straightline knowledge soundness* has essentially the same security as the underlying SNARK (i.e., recursive composition incurs essentially no security loss).

We describe how straightline knowledge soundness is achieved by SNARKs in several oracle models, which results in a highly efficient security analysis of PCD that makes black-box use of the SNARK's oracle (there is no need to instantiated the oracle to carry out the security reduction).

As a notable application, our work offers an idealized model that provides new, albeit heuristic, insights for the concrete security of *recursive STARKs* used in blockchain systems. Our work could be viewed as partial evidence justifying the parameter choices for recursive STARKs made by practitioners.

Keywords: proof-carrying data · succinct non-interactive arguments · relativization · concrete security

1 Introduction

Proof-carrying data (PCD) [21] is a powerful cryptographic primitive that allows mutually distrustful parties to perform distributed computation in an efficiently verifiable manner. PCD generalizes the notion of incrementally-verifiable computation (IVC) [45], and has found applications in enforcing language semantics [26], verifiable

E. Boyle and M. Mahmoody (Eds.): TCC 2024, LNCS 15365, pp. 464–496, 2025.
https://doi.org/10.1007/978-3-031-78017-2_16

MapReduce computations [22], image authentication [37], verifiable registries [44], privacy pools [3], blockchains [10,18,32,38], and more.

Known PCD constructions (and practical IVC constructions) are obtained via *recursive proof composition*, a framework for building PCD from simpler primitives such as SNARKs [7,8,20] or accumulation schemes [9,11–14,33,34].[1] Constructions differ, but the high-level idea is similar: to prove the correctness of t computation steps given a correctness proof for $t-1$ steps, one proves that "step t is correct *and* there is a valid proof for the first $t-1$ steps".

There are several practically efficient constructions of PCD, which has sparked keen industry interest and led to real-world deployments [35,40,41,43]. However, the concrete cost of the security reduction from PCD to the underlying primitive is not well understood: *there are no comprehensive guidelines for securely instantiating PCD constructions*. In fact, an initial motivation for this work was the desire to better understand the concrete security of recursive STARKs [43] used in blockchain systems.

The prevailing practice in real-world deployments is setting parameters so that the underlying SNARK or accumulation scheme achieves the desired security level, and then assuming that the resulting PCD construction inherits the same security level. *But this fails to account for the potential security loss in the security reduction from PCD to the primitive(s) underlying its construction.*

This state of affairs leads us to ask a basic question:

What is the concrete security cost of recursive proof composition?

Known Security Analyses. The security analysis of most PCD constructions works only for a constant number of recursions ([7–9,11–14,20,33,34]). Informally, this is because the security reduction recursively invokes an underlying knowledge extractor that, at each invocation, incurs a polynomial blowup in time/size relative to the prior invocation. Moreover, in many settings, this blow up is unknown because it originates from an underlying knowledge assumption (e.g., a knowledge-of-exponent assumption). Overall this state of affairs implies that one is unable to set security parameters, and that the security loss is exponential in the recursion depth. This is *considerably worse* than "the security of PCD is approximately that of the underlying primitive" (the prevailing practice).

What About Oracle Models? The aforementioned inefficiencies of knowledge extraction generally do not arise for SNARKs (or accumulation schemes) constructed in oracle models. This is because the knowledge extractor is explicit (it is constructed rather than assumed), and deduces a witness merely by analyzing the prover's queries to the oracle and their answers; this does not require any access to the prover itself, and avoids rerunning the prover multiple times (which incurs significant time or error overheads). Unfortunately, PCD constructions typically make a non-black-box use of the underly-

[1] A separate line of work constructs IVC for deterministic computations from falsifiable cryptographic assumptions using different tools (see [39] and references therein). These elegant constructions are less relevant to the motivation of this paper as, typically, applications of PCD and IVC require supporting nondeterministic computations.

ing SNARK or accumulation scheme,[2] which requires instantiating the oracle (a heuristic step), and so the security reduction cannot take advantage of the efficient knowledge extraction previously available in the oracle model. Instead, the security reduction assumes some (non-black-box) knowledge extractor for the heuristically derived scheme.

Hope: Black-Box Constructions of PCD. Several works construct PCD in oracle models *without* instantiating the oracle (that is, while making a black-box use of the oracle) [16,17,21]. The key step is obtaining a *relativized SNARK*, a SNARK in an oracle model that can prove computations that themselves involve calls to the oracle. Then recursive proof composition can be used to directly obtain PCD in the same oracle model, via a security analysis that involves an explicit extractor.

1.1 Our Results

In this paper we show that PCD constructions obtained from SNARKs with *straightline knowledge soundness* have essentially the same security as the underlying SNARK. Afterwards, we explain how this setting arises in several constructions of interest, including in deployed systems. In particular, our work gives partial justification for parameter settings currently used by practitioners in certain deployed PCD constructions.[3]

PCD from Straightline Extraction. Suppose that we are given a relativized SNARK in a certain oracle model. The canonical construction of PCD from a SNARK in the standard model [8] straightforwardly extends to the relativized case. Indeed, the SNARK prover can prove the correctness of oracle computations, and in particular can prove the correctness of the SNARK verifier (which queries the oracle).

We show that if the relativized SNARK has a straightline extractor then the resulting PCD scheme has a straightline extractor with the same error as the underlying SNARK.

Theorem 1 (informal). *Let* ARG *be a relativized non-interactive argument in an oracle model, and let* PCD *be the PCD scheme obtained from* ARG *via the canonical construction (adapted to the relativized setting).*

Suppose that ARG *has a straightline extractor with knowledge soundness error* $\kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}, \mathsf{s})$ *and extraction time* $\mathsf{t}_{\mathsf{ARG}}(\lambda, \mathsf{q})$*, where* $\lambda \in \mathbb{N}$ *is the security parameter,* $\mathsf{q} \in \mathbb{N}$ *is the number of queries by the adversary to the oracle, and* s *is the size of the adversary. Then* PCD *has a straightline extractor with:*

- *knowledge soundness error* $\kappa_{\mathsf{PCD}}(\lambda, \mathsf{q}, \mathsf{s}, N) \leq \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}, \mathsf{s}')$ *where* $\mathsf{s}' := \mathsf{s} + O(N \cdot \mathsf{t}_{\mathsf{ARG}}(\lambda, \mathsf{q}))$*, and*
- *extraction time* $\mathsf{t}_{\mathsf{PCD}}(\lambda, \mathsf{q}, N) \leq O(N \cdot \mathsf{t}_{\mathsf{ARG}}(\lambda, \mathsf{q}))$.

[2] The statement that "there exists a valid proof" refers to the *verifier* of the underlying SNARK or accumulation scheme. As such, the resulting PCD scheme makes non-black-box use of the verifier for the underlying scheme.

[3] There are other PCD constructions of practical interest that do not fit our setting (e.g., those based on knowledge-of-exponent assumptions). Achieving security reductions that yield useful concrete security bounds for these remains an open problem.

Above, N is the maximum number of nodes in the PCD distributed computation.

Above, the additive term $O(N \cdot \mathsf{t}_{\mathsf{ARG}}(\lambda, \mathsf{q}))$ intuitively corresponds to the N extractions required to produce a PCD distributed computation of size at most N. Note that, since extraction is straightline, the extraction times $\mathsf{t}_{\mathsf{ARG}}$ and $\mathsf{t}_{\mathsf{PCD}}$ do not depend on adversary size.

We discuss two applications of Theorem 1: in Sect. 1.1 we discuss an application the black-box PCD constructions; and in Sect. 1.1 we discuss an application to partially justify parameter settings used in certain deployed PCD constructions.

Application: Black-Box PCD Constructions. Several works [16,17,21] construct PCD in oracle models, with black-box security reductions to falsifiable cryptographic assumptions.[4] While [17] constructs a rewinding knowledge extractor (and leaves open the question of constructing a straightline knowledge extractor), [16,21] construct straightline knowledge extractors in their respective oracle models. These latter works roughly achieve the following: they construct a relativized SNARK with a certain (straightline) knowledge soundness error κ_{ARG} and then show that the resulting PCD scheme has (straightline) knowledge soundness error (roughly) $\kappa_{\mathsf{PCD}}(\lambda, \mathsf{q}, \mathsf{s}, N) \leq N \cdot \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}, \mathsf{s}')$, where $\mathsf{s}' := \mathsf{s} + O(N \cdot \mathsf{t}_{\mathsf{ARG}}(\lambda, \mathsf{q}))$.

Our Theorem 1 offers a significant improvement for [16,21]: the knowledge soundness error $\kappa_{\mathsf{PCD}}(\lambda, \mathsf{q}, \mathsf{s}, N) \leq \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}, \mathsf{s}')$, *eliminating the multiplicative factor* N (PCD distributed computation size). We suspect that this upper bound is tight, as the only overhead comes from increasing the adversary size from s to $\mathsf{s}' = \mathsf{s} + O(N \cdot \mathsf{t}_{\mathsf{ARG}}(\lambda, \mathsf{q}))$, which reflects the additive cost to recover a PCD distributed computation of size N by invoking the SNARK extractor (whose running time is $\mathsf{t}_{\mathsf{ARG}}(\lambda, \mathsf{q})$) for N times.

Application: Hash-Based PCD. Hash-based SNARKs have found widespread deployment in practice. Security parameters of such SNARKs are heuristically set according to the random oracle methodology [4]: first, model the hash function as a random function (even though it is not), which results in a SNARK in the ROM that "idealizes" the given hash-based SNARK; second, establish concrete security bounds in the ROM; finally, set security parameters of the hash-based SNARK according to the analysis for the SNARK in the ROM.

The random oracle methodology applied to the hash-based SNARK is thus tantamount to a *conjecture*: attacks against the hash-based SNARK are no more effective than attacks against the corresponding SNARK in the ROM (hence it is reasonable to set security parameters of the former according to the latter). Such conjectures are generally believed to hold for "natural" cryptographic protocols that use hash functions.[5]

[4] Such reductions are unlikely to exist in the standard model [29], as PCD can be used to construct a SNARK [8].

[5] The random oracle methodology is widely used across cryptography to set the security parameters of protocols that rely on cryptographic hash functions (and possibly other cryptographic building blocks). The methodology must, nevertheless, be applied with caution because it does not work for every protocol [15].

In certain applications, the hash-based SNARK is recursively composed, leading to deployments of hash-based PCD constructions. Known security analyses of these PCD constructions incur expensive blowups, which practitioners have disregarded as those security analyses would lead to setting parameters that are prohibitively expensive. In other words, the common practice is to set security parameters as if the security reduction incurred no costs. Below we elaborate on these challenges, and then we propose how the results in this paper can be viewed as providing partial justification for current parameter settings in practice.

Challenges. Once the SNARK in the ROM is heuristically instantiated as a hash-based SNARK, we lose the explicit knowledge extractor constructed in the ROM. Hence, to analyze the resulting hash-based PCD construction, prior work postulates the existence of a non-black-box knowledge extractor for the hash-based SNARK in the standard model. But such a knowledge extractor is weak (it leads to a PCD knowledge extractor whose time/size has an exponential dependence on the recursion depth), and also rules out any hope for concrete security because we know nothing of the postulated knowledge extractor.

Alternatively, why not apply the random oracle methodology? Idealize the hash-based PCD construction as a PCD in the ROM, then set security parameters of the hash-based PCD according to a concrete security analysis of the corresponding PCD in the ROM. Unfortunately, this is problematic because it would require the underlying SNARK in the ROM to be relativized; indeed, the SNARK prover would have to attest to computations involving the random oracle (namely, its own SNARK verifier). However known SNARKs in the ROM are not relativized and, in fact, relativized SNARKs in the ROM do not exist [1].[6]

Our Proposal. We propose a method, based on Theorem 1, that provides new insights into the concrete security cost of PCD obtained from hash-based SNARKs. Specifically, we *can* idealize the hash-based PCD construction in a less straightforward way, resulting in a PCD construction in the ROM that, albeit not succinct, is covered by Theorem 1 and could be reasonably conjectured to capture the security of the original hash-based PCD construction. Of course, the security of hash-based PCD constructions merits further study beyond this work, given the delicate nature of heuristic instantiations of SNARKs in the ROM [2].

Briefly, the hash-based PCD construction makes a *specific* non-black-box use of the underlying hash function: it uses the underlying hash-based SNARK to prove correct execution of computations that involve the hash function itself. In the idealization, we can model this as a non-succinct SNARK in the ROM where query-answer pairs to the random oracle of the proved computation are simply included in the argument string as claims to be checked directly by the verifier (which has access to the oracle). This relativized "non-succinct NARK" in the ROM directly leads to a (non-succinct) PCD construction in the ROM, whose concrete security follows from Theorem 1. This latter PCD construction in the ROM closely models the original hash-based PCD construc-

[6] Relativization is distinct from other limitations of SNARKs in the presence of oracles: [28] studies limitations of knowledge extraction for adversaries that access oracles exogenous to the SNARK scheme itself (e.g., the signing oracle of a signature scheme).

tion, and, in analogy to the random oracle methodology, one may conjecture that attacks against the hash-based PCD construction are no more effective than attacks against the idealized (non-succinct) PCD construction in the ROM sketched above.

The above steps are summarized in Fig. 1.

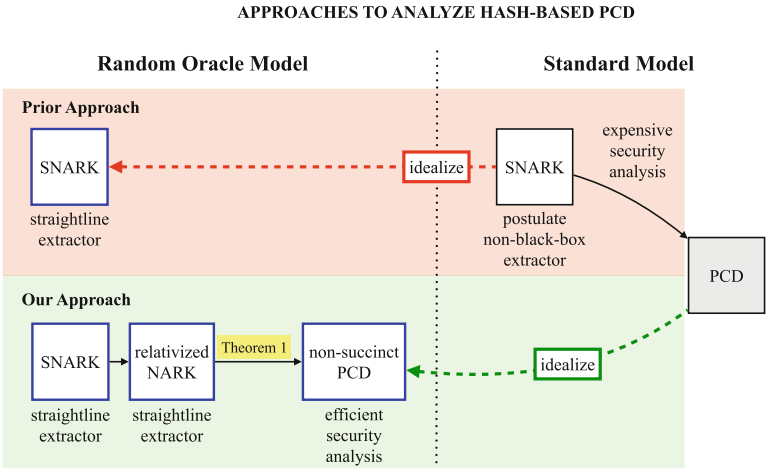

Fig. 1. The grey box "PCD" on the right represents the hash-based PCD construction used in practice whose concrete security we wish to understand. *Top:* Prior work provides an expensive security analysis based on a hash-based SNARK, whose security is heuristically set by equating it to a corresponding idealized SNARK in the ROM (i.e., via the random oracle methodology). *Bottom:* We directly idealize the hash-based PCD construction, equating its security to a corresponding (non-succinct) PCD in the ROM whose security we establish. (Color figure online)

In sum, our Theorem 1 provides new insights for practitioners: in the above heuristic sense (inspired by the random oracle methodology), one may conjecture that the concrete security of hash-based PCD constructions equals that of the underlying idealized SNARK in the ROM, matching widespread practices for recursive proof composition (and thus providing some justification for these practices).

See Sect. 2.3 for more discussion.

A-Priori Unknown N. In the (pure) ROM setting described above, the size of the adversary does not matter (only the query bound matters). Hence the knowledge soundness error simplifies to

$$\kappa_{\mathsf{PCD}}(\lambda, \mathsf{q}, N) \leq \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q})$$

which is independent of N.[7] Therefore, with our improvement, suitably setting κ_{ARG} once suffices for *all* distributed computations (which may have arbitrarily large size that is unknown a priori), whereas with the prior results one would have to set κ_{ARG}

[7] The extraction time is as before and, necessarily, depends on N, as the extractor outputs a distributed computation of size N.

depending on the pre-specified bound N on the size of the distributed computation. In Sect. 2.4 we describe a natural real-world example where there does not exist any pre-specified bound N on the size of a valid distributed computation.

2 Techniques

We overview the main ideas underlying our results. In Sect. 2.1 we discuss our improved security analysis for PCD constructed from relativized SNARKs with straightline extractors. In Sect. 2.2 we discuss how our improvement applies to prior relativized SNARKs in different oracle models. In Sect. 2.3 we discuss relativized "non-succinct NARKs" in the random oracle model, and implications to real-world constructions. In Sect. 2.4 we discuss a real-world compliance predicate that did not have security guarantees (in terms of knowledge soundness) prior to our work. In Sect. 2.5 we discuss how our security analysis extends to the case of straightline extractors that are probabilistic and may query the oracle.

2.1 Security Analysis of PCD from Straightline Extractors

We elaborate on Theorem 1 and outline the main ideas of its proof. The technical details that make these discussions precise are provided in Sects. 4 and 5.

Review: PCD. Proof-carrying data (PCD) is a cryptographic primitive that enables untrusted provers to efficiently demonstrate the correctness of a distributed computation. A distributed computation T is viewed as a directed acyclic graph in which each vertex is labeled with *local data* and each edge is labeled with a *message*; the computation *output* is the message on the lexicographically-first edge into a sink. Correctness is determined by a given *compliance predicate* ϕ: T is ϕ-compliant if, for every vertex in T, ϕ outputs 1 when given as input the vertex's output message, local data, and input messages. The *transcript size* and *transcript depth* of ϕ are the largest size and largest depth of any ϕ-compliant distributed computation T.[8] A *PCD scheme* is a tuple $\mathsf{PCD} = (\mathbb{P}, \mathbb{V})$ for proving/verifying ϕ-compliance of distributed computations, as follows.

- The PCD prover $\mathbb{P}$ receives an output message z, local data w_{loc}, and input messages $(z_i)_i$ together with PCD proofs $(\Pi_i)_i$ (each proof Π_i attests to the ϕ-compliance of the corresponding message z_i), and produces a PCD proof Π for the ϕ-compliance of the output message z.
- The PCD verifier $\mathbb{V}$ receives a message z and PCD proof Π, and outputs a decision bit.

The PCD scheme is complete if proofs for compliant messages are accepted by the PCD verifier. The PCD scheme is knowledge sound if every malicious PCD prover producing a message and proof accepted by the PCD verifier "knows" a compliant

[8] In particular, the transcript size and transcript depth may or may not be bounded for a particular compliance predicate ϕ.

distributed computation whose output is that message, up to some error. This error is bounded by a knowledge soundness error function $\kappa_{\mathsf{PCD}}(\lambda, \mathsf{q}, \mathsf{s}, N)$, which depends on the security parameter λ, number of queries q by the adversary to the oracle, size s of the adversary, largest size N of any ϕ-compliant distributed computation, and other parameters that we omit here for simplicity. Our Theorem 1 establishes an improved bound on the knowledge soundness error $\kappa_{\mathsf{PCD}}(\lambda, \mathsf{q}, \mathsf{s}, N)$ of PCD schemes obtained from non-interactive arguments with straightline knowledge soundness. See Sect. 3.2 for a formal definition of a PCD scheme.

Limitations of PCD from SNARKs in the Standard Model. A PCD scheme in the standard model can be constructed from any SNARK (with adaptive security) in the standard model [8]. Informally, the PCD prover uses the SNARK prover to produce a short proof attesting that (i) the compliance predicate accepts the output message, local data, and input messages, and (ii) input messages carry valid SNARK proofs; the PCD verifier uses the SNARK verifier to check the proof accompanying a message.

However, the security analysis works only for compliance predicates with *constant* transcript depth. This is because SNARKs in the standard model satisfy a modest notion of adaptive knowledge soundness: *non-black-box knowledge soundness*. Informally, for every SNARK adversary there exists a knowledge extractor, whose size is polynomially-related to the adversary size, such that, whenever the SNARK adversary convinces the SNARK verifier, the knowledge extractor outputs a valid witness (up to some error). If the polynomial blowup from adversary to extractor is $n \mapsto n^c$ then the security reduction for a PCD prover of size n would yield a PCD extractor of size (roughly) n^{c^d}, where d is the transcript depth of the compliance predicate.[9] This size blowup is huge in concrete terms, and asymptotically this requires d to be constant.

PCD from Relativized SNARKs. In the *relativized setting*, we consider SNARKs that are constructed in an idealized model where the SNARK can prove/verify computations involving *the same oracle*. In other words, all (honest and malicious) parties have access to an oracle sampled according to a certain distribution and, in particular, the SNARK prover and SNARK verifier may query the oracle; crucially, the SNARK is required to work even for relations that are defined relative to the same oracle. See Sect. 3.1 for a formal definition of a relativized non-interactive argument in an oracle model.

The aforementioned canonical PCD construction in the standard model extends naturally to the relativized setting. Indeed, in the recursive step of the construction, the PCD prover produces a SNARK proof attesting the computation of the SNARK verifier, which in turn involves oracle calls. A relativized SNARK prover possesses the capability to generate such a proof. Furthermore, the security analysis of the relativized PCD construction can be carried over but presents the same blowup encountered in the standard model.

Enabler: Straightline Extraction. In the relativized setting, we have the additional benefit of a stronger knowledge soundness property: many SNARKs in oracle models have a (universal) *straightline extractor*, a notion that is uniquely defined within an oracle model and lacks an equivalent counterpart in the standard model. A straightline

[9] The dependence is on transcript depth rather than transcript size because the security reduction simultaneously extracts from all SNARK proofs at the same transcript depth (see, e.g., [20]).

extractor does *not* get access to the malicious SNARK prover; instead, it produces a witness by examining the following: the instance and argument string output by the malicious SNARK prover, the sequence of queries to the oracle performed by the malicious SNARK prover, and the corresponding query answers. We refer to the list of query-answer pairs as the *query-answer trace* tr of the SNARK prover.

Definition 1 (Informal). $\mathsf{ARG} = (\mathcal{P}, \mathcal{V})$ *for a relation* R *has* **straightline knowledge soundness error** κ_{ARG} *if there exists a polynomial-time deterministic extractor* $\mathcal{E}$ *such that, for every security parameter* $\lambda \in \mathbb{N}$ *and* q*-query* s*-size prover* $\widetilde{\mathcal{P}}$*,*

$$\Pr\left[\begin{array}{l|r} (\mathbb{x}, \mathbb{w}) \notin R & f \leftarrow \mathcal{U}(\lambda) \\ \wedge\, \mathcal{V}^f(\mathbb{x}, \pi) = 1 & (\mathbb{x}, \pi) \xleftarrow{\mathsf{tr}} \widetilde{\mathcal{P}}^f \\ & \mathbb{w} \leftarrow \mathcal{E}(\mathbb{x}, \pi, \mathsf{tr}) \end{array}\right] \leq \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}, \mathsf{s}) \ .$$

Above tr *denotes the query-answer trace of* $\widetilde{\mathcal{P}}$ *with the oracle* f*.*

The running time of the straightline extractor $\mathcal{E}$ does *not* depend on the running time of the malicious SNARK prover, but only on the number of query-answer pairs in the SNARK prover's trace tr (as well as the instance $\mathbb{x}$ and SNARK proof π). Straightline extractors are common in oracle models, and as we explain shortly *they will enable us to avoid the blowup in the PCD extractor size discussed above*.

We begin by sketching a (straightline) PCD extractor $\mathbb{E}$ that is naturally obtained from the given straightline SNARK extractor $\mathcal{E}$, by recursively extracting prior messages (and SNARK proofs), one vertex at a time. The straightline extractor $\mathbb{E}$ receives as input the compliance predicate ϕ, the message z_{out} and proof $\boldsymbol{\Pi}_{\text{out}}$ output by the malicious PCD prover, and the query-answer trace tr of the malicious PCD prover; $\mathbb{E}$ aims to output a ϕ-compliant PCD transcript T whose output is z_{out}.

$\mathbb{E}(\phi, z_{\text{out}}, \boldsymbol{\Pi}_{\text{out}}, \mathsf{tr})$:

1. Initialize a PCD transcript T as an empty graph.
2. Add to T vertices v_0 and v_1, and add to T the edge (v_1, v_0) with label $(z_{\text{out}}, \boldsymbol{\Pi}_{\text{out}})$.
3. Initialize an extraction queue $\mathcal{L}$ with the vertex v_1.
4. While the extraction queue $\mathcal{L}$ is not empty:
   (a) Pop the first vertex v from the queue $\mathcal{L}$.
   (b) Let $(z, \boldsymbol{\Pi})$ be the label of the unique outgoing edge of v.
   (c) Run the SNARK knowledge extractor $\mathcal{E}((\phi, z), \boldsymbol{\Pi}, \mathsf{tr})$ to obtain a witness $\mathbb{w}$.
   (d) Parse $\mathbb{w}$ to obtain local data w_{loc} and input messages and proofs $((z_i, \boldsymbol{\Pi}_i))_i$ for v.
   (e) Label v the vertex by w_{loc}.
   (f) For each message-proof pair $(z_i, \boldsymbol{\Pi}_i)$: add a new child vertex of v, label the new edge with $(z_i, \boldsymbol{\Pi}_i)$, and add the new vertex to the extraction queue $\mathcal{L}$.
5. Output the PCD transcript T.

For the rest of this section, let λ be the security parameter, q an upper bound on the number of oracle queries made by the malicious PCD prover $\widetilde{\mathbb{P}}$, s an upper bound on the

size of the malicious PCD prover, and N an upper bound on the size of any ϕ-compliant transcript.

Security Analysis Inspired by Prior Work. Security analyses of PCD in prior works based on straightline extractors [16,21] bound the knowledge error (roughly) as $\kappa_{\mathsf{PCD}}(\lambda, \mathsf{q}, \mathsf{s}, N) \leq N \cdot \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}, \mathsf{s}')$ where $\mathsf{s}' := \mathsf{s} + O(N \cdot \mathsf{t}_{\mathsf{ARG}}(\lambda, \mathsf{q}))$. Intuitively, each recursion incurs the SNARK knowledge soundness error of $\kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}, \mathsf{s}')$. In more detail, the i-th extraction is achieved by invoking the SNARK knowledge extractor for a corresponding i-th SNARK prover $\widetilde{\mathcal{P}}_i^f$, which outputs the message and proof in the label of the outgoing edge of the i-th vertex considered by $\mathbb{E}$. Note that $\widetilde{\mathcal{P}}_i^f$ runs in time $\mathsf{s} + O(i \cdot \mathsf{t}_{\mathsf{ARG}}(\lambda, \mathsf{q}))$ because, to perform the extraction associated to the i-th vertex, it first has to perform the extractions associated to the first $i - 1$ vertices (as $\mathbb{E}$ does). Using a union bound, the success probability of the PCD adversary can be upper bounded by the sum of the success probabilities of all these argument adversaries (one per vertex in the transcript), yielding the aforementioned upper bound $\kappa_{\mathsf{PCD}}(\lambda, \mathsf{q}, \mathsf{s}, N) \leq N \cdot \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}, \lambda, \mathsf{q}, \mathsf{s}')$.

This bound (obtained via straightline knowledge extraction) is a significant improvement over the exponential blow-up incurred when only relying on non-black-box knowledge extraction, and as discussed, is achieved by prior works on PCD in oracle models [16,21]. However, the multiplicative factor of N impacts concrete security, and, in fact, is unacceptable when N is unknown a priori. (There are compliance predicates deployed in the real world for which N is unknown a priori; see Sect. 2.4.)

Our Security Analysis. We improve the security analysis of PCD from straightline knowledge extraction: we avoid paying for the multiplicative factor of N, bounding the PCD knowledge soundness error as $\kappa_{\mathsf{PCD}}(\lambda, \mathsf{q}, \mathsf{s}, N) \leq \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}, \mathsf{s}')$ where $\mathsf{s}' := \mathsf{s} + O(N \cdot \mathsf{t}_{\mathsf{ARG}}(\lambda, \mathsf{q}))$. Intuitively, we force the SNARK adversary to pinpoint the problematic vertex (if any) in a PCD distributed computation transcript T by running the PCD adversary $\widetilde{\mathbb{P}}$ *once*.

In particular, our SNARK adversary $\widetilde{\mathcal{P}}$ follows the PCD knowledge extractor $\mathbb{E}$. If the PCD transcript T extracted by $\mathbb{E}$ is not compliant with the predicate ϕ, there must exist at least one problematic vertex in T. The SNARK adversary $\widetilde{\mathcal{P}}$ reconstructs T and searches for this problematic vertex along the way.

$\widetilde{\mathcal{P}}^f$:

1. Run the PCD adversary $\widetilde{\mathbb{P}}^f$ to obtain its output $(\phi, z_{\mathrm{out}}, \boxplus_{\mathrm{out}})$, and its query-answer trace tr.
2. Initialize a PCD transcript T as an empty graph.
3. Add to T vertices v_0 and v_1, and add to T the edge (v_1, v_0) with label $(z_{\mathrm{out}}, \boxplus_{\mathrm{out}})$.
4. Initialize an extraction queue $\mathcal{L}$ with the vertex v_1.
5. While $\mathcal{L}$ is not empty:
   (a) Pop the first vertex v from the queue $\mathcal{L}$.
   (b) Let $(z, \boxplus)$ be the label of the unique outgoing edge of v.
   (c) Run the SNARK knowledge extractor $\mathcal{E}((\phi, z), \boxplus, \mathsf{tr})$ to obtain a witness $\mathbb{w}$.
   (d) Parse $\mathbb{w}$ to obtain local data w_{loc} and input messages and proofs $((z_i, \boxplus_i))_i$ for v.

(e) For each message-proof pair $(z_i, \text{𝕄}_i)$: add a new child vertex of v, label the new edge with $(z_i, \text{𝕄}_i)$, and add the new vertex to the extraction queue $\mathcal{L}$.
(f) If at least one of following is true, output $(z, \text{𝕄})$:
   i. $\phi(z, w_{\text{loc}}, (z_i)_i) \neq 1$ (i.e., v is not ϕ-compliant).
   ii. There exists i such that $\mathcal{V}^f(z_i, \text{𝕄}_i) \neq 1$
   (i.e., the SNARK verifier rejects $(z_i, \text{𝕄}_i)$).
6. If no output so far, output an arbitrary message-proof pair.

The malicious SNARK prover $\widetilde{\mathcal{P}}$ above follows the PCD extractor $\mathbb{E}$, with additional checks in Item 5f. If the PCD transcript T is not ϕ-compliant, there must be at least one vertex v in T such that the outgoing message of v is inconsistent with the incoming messages to v (and the local data at v), and the SNARK verifier does not catch this. In other words, computation at vertex v successfully fools the SNARK verifier. The label corresponding to the outgoing edge of the first such v (in the order $\mathbb{E}$ extracts) is output by $\widetilde{\mathcal{P}}$.

The number of queries made by $\widetilde{\mathcal{P}}$ equals the number of queries made by $\widetilde{\mathbb{P}}$, plus the additional queries by the SNARK verifier $\mathcal{V}$ in Item 5(f)ii (which is invoked at most N times). In fact, in the technical sections, we explain how to avoid this latter additive cost, observing that it suffices to run only a "part" of the SNARK verifier $\mathcal{V}$ that does not query the oracle (see details in Sect. 5).

Lastly, the size of $\widetilde{\mathcal{P}}$ is the sum of the size s of $\widetilde{\mathbb{P}}$, the size of the N invocations of the argument extractor $\mathcal{E}$ (and some processing in between), all of which is upper bounded by $\mathsf{s} + O(N \cdot \mathsf{t}_{\mathsf{ARG}}(\lambda, \mathsf{q}))$.

We conclude $\kappa_{\mathsf{PCD}}(\lambda, \mathsf{q}, \mathsf{s}, N) \leq \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}, \mathsf{s}')$ as desired.

The Preprocessing Setting. For wider applicability of our results, in the technical sections, we work in a more general setting. We consider SNARKs (and PCD) in the *preprocessing model*, which means that an additional algorithm known as the *indexer* may do an offline computation on the "offline" part of the instance, producing a corresponding proving key and verification key to be used for proving and verifying proofs.

2.2 Application: Improved Concrete Security for Black-Box PCD Constructions

We discuss two oracle models from prior work where one can construct relativized SNARKs with straightline knowledge soundness. Our Theorem 1 yields a security bound for PCD schemes obtained in these oracle models that is a significant improvement over previously-known bounds.

Arithmetized Random Oracle. [16] constructs PCD in the *arithmetized random oracle model* (AROM), which is an idealization of capabilities associated to the arithmetization of a hash function. In this model, all parties have access to a random oracle, which as usual can be viewed as the idealization of some concrete hash function h; in addition, all parties have access to an associated arithmetization oracle, which can be viewed as an idealization of a low-degree polynomial p_h that "encodes" the circuit of h.

Briefly, [16] shows that queries to the AROM can be "accumulated", and they show how this implies that any SNARK in the ROM can be transformed into a relativized

SNARK in the AROM; moreover, the relativized SNARK has a straightline extractor if the given SNARK has a straightline extractor. In turn, this implies a construction of PCD in the AROM (that makes a black-box use of the AROM).

The analysis in [16] implies an error bound for PCD that is (roughly) $\kappa_{\mathsf{PCD}}(\lambda, \mathsf{q}, \mathsf{s}, N) \leq N \cdot \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}, \mathsf{s} + O(N \cdot \mathsf{t}_{\mathsf{ARG}}(\lambda, \mathsf{q})))$ where κ_{ARG} is the (straightline) knowledge soundness error of the underlying relativized SNARK in the AROM. Our Theorem 1 improves the error bound for PCD to $\kappa_{\mathsf{PCD}}(\lambda, \mathsf{q}, \mathsf{s}, N) \leq \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}, \mathsf{s} + O(N \cdot \mathsf{t}_{\mathsf{ARG}}(\lambda, \mathsf{q})))$.

Signed Random Oracle. [21] constructs PCD in an oracle model that combines the random oracle model and a signature scheme, which here we refer to as the *signed random oracle model* (SROM). All parties have access to an oracle that, on a new input x, samples a random answer y, generates a signature σ on (x, y) under a secret signing key embedded in the oracle, and outputs (y, σ); repeated inputs have the same answers.

Intuitively, this model facilitates a PCD construction because the SNARK verifier does not need to query the oracle: to check that the oracle answers x with (y, σ), one can verify that σ is a valid signature on the message (x, y) using the oracle's public key; there is no need to query the oracle at x.

More generally, any SNARK in the ROM (with straightline extraction) directly implies a relativized SNARK in the SROM (with straightline extraction), up to an error that depends on the security of the signature scheme. To prove an oracle computation, invoke the prover of the SNARK in the ROM on the computation where all oracle calls are replaced with sub-computations that verify signatures on the relevant messages; to verify the corresponding SNARK proof, invoke the verifier of the SNARK in the ROM.

The aforementioned relativized SNARK implies a corresponding PCD construction in the SROM (which is essentially the one studied in [21] but reinterpreted through the relativization lens). The analysis in [21] implies an error bound that is (roughly) $\kappa_{\mathsf{PCD}}(\lambda, \mathsf{q}, \mathsf{s}, N) \leq N \cdot \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}, \mathsf{s} + O(N \cdot \mathsf{t}_{\mathsf{ARG}}(\lambda, \mathsf{q})))$ where κ_{ARG} is the (straightline) knowledge soundness error of the underlying relativized SNARK in the SROM. Our Theorem 1 improves this error bound to $\kappa_{\mathsf{PCD}}(\lambda, \mathsf{q}, \mathsf{s}, N) \leq \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}, \mathsf{s} + O(N \cdot \mathsf{t}_{\mathsf{ARG}}(\lambda, \mathsf{q})))$.

Remark 1. [17] constructs PCD in the *low-degree random oracle model (LDROM)*, where all parties have access to a random low-degree extension of a random oracle. Specifically they construct a relativized SNARK in the LDROM, and from there obtain PCD in the LDROM. However, the relativized SNARK in [17] is only shown to have a rewinding extractor, and because of this they show security of the PCD construction only for compliance predicates with constant transcript depth. Constructing a straightline extractor for the relativized SNARK in the LDROM in [17] (or, indeed, any relativized SNARK in the LDROM) remains an open problem, which precludes our Theorem 1 from use in the LDROM setting.

2.3 Application: a Paradigm to Set Security for Hash-Based PCD

Relativized SNARKs in the ROM do not exist [1]. Nevertheless, one can achieve a weak form of relativized SNARKs in the ROM that implies a corresponding weak form

of PCD in the ROM (using no assumptions or heuristics), for which our Theorem 1 gives concrete security bounds. This weak form of PCD in the ROM can be (heuristically) viewed as an idealization of an important class of (succinct) hash-based PCD constructions used in practice. In turn, we gain new insights into the concrete security of these hash-based PCD constructions. We elaborate on this below.

"Weak" Relativized SNARKs in the ROM. One can construct relativized SNARKs in the ROM for relations decidable via computations that perform few queries to the random oracle. The construction below remains secure regardless of the number of queries. However, if the number of queries to the oracle is large, then the resulting argument system is a relativized "non-succinct NARK" rather than a relativized SNARK.

Suppose we have a non-relativized SNARK in the ROM [6,23–25,36] with proof size ℓ. Consider an oracle relation $R^{\mathcal{U}}$ whose decision involves q queries to the random oracle. We can construct a SNARK in the ROM for $R^{\mathcal{U}}$ with proof size $\ell + O(\mathsf{q} \cdot \lambda)$, where λ is the output size of the random oracle. We modify the circuit that checks whether a given instance-witness pair is in the relativized relation: remove each oracle gate and instead read a corresponding query-answer pair from an augmented instance (which now additionally stores the list of all query-answer pairs for the computation). The new circuit is proved using the given non-relativized SNARK in the ROM; and the resulting SNARK proof of size ℓ is accompanied by the list of query-answer pairs, increasing its size to $\ell + O(\mathsf{q} \cdot \lambda)$. The new SNARK verifier checks the SNARK proof and checks that the list of query-answer pairs is consistent with the random oracle.

Say that the non-relativized SNARK $(\mathcal{P}_1, \mathcal{V}_1)$ is for the circuit satisfiability relation R_{CSAT}. We construct a (weak) relativized SNARK $(\mathcal{P}_2, \mathcal{V}_2)$ for the oracle relation $R^f_{\mathrm{CSAT}} \coloneqq \{(C, \mathbb{x}, \mathbb{w}) : C^f(\mathbb{x}, \mathbb{w}) = 1\}$.

- $\mathcal{P}_2^f(C, \mathbb{x}, \mathbb{w})$:
  1. Run $C^f(\mathbb{x}, \mathbb{w})$ to obtain its query-answer trace tr_C.
  2. Construct the new (non-oracle) circuit C' that, on input $((\mathbb{x}, \mathsf{tr}_C), \mathbb{w})$, computes $C(\mathbb{x}, \mathbb{w})$ by answering C's queries to f with the query-answer pairs in tr_C.
  3. Run the SNARK prover for R_{CSAT}: $\pi \leftarrow \mathcal{P}_1^f(C', (\mathbb{x}, \mathsf{tr}_C), \mathbb{w})$.
  4. Output (π, tr_C).
- $\mathcal{V}_2^f(C, \mathbb{x}, (\pi, \mathsf{tr}_C))$:
  1. Construct the new (non-oracle) circuit C' from C like $\mathcal{P}_2$ does.
  2. Check that $\mathcal{V}_1^f(C', (\mathbb{x}, \mathsf{tr}_C), \pi) = 1$.
  3. Check that tr_C is consistent with f (by directly querying f for each query in tr_C).

The security of the SNARK for R^f_{CSAT} follows from the security of the SNARK for R_{CSAT}. Specifically, the transformation *preserves straightline extraction*: if the SNARK for R_{CSAT} has a straightline extractor (with a knowledge error),[10] then so does the SNARK for R^f_{CSAT} constructed above (with the same knowledge error).

"Weak" PCD in the ROM. The weak relativized SNARK $(\mathcal{P}_2, \mathcal{V}_2)$ in the ROM directly leads to a weak PCD scheme $(\mathbb{P}, \mathbb{V})$ in the ROM. The PCD construction invokes the

[10] Achieving straightline extraction in the ROM is straightforward; see prior work [6,23–25,36].

SNARK for relations that involve the SNARK verifier, which makes a small number of queries to the random oracle. Hence the above relativized SNARK can be used to recursively prove the correctness of the SNARK verifier. Our Theorem 1 provides a bound on the knowledge soundness error of the resulting PCD scheme, thanks to the straightline knowledge soundness of the SNARK. However, with each recursive step, proof size increases, leading to a PCD construction that is *not succinct*, aligning with the limitations of PCD in the ROM [1,19,31].

The Silver Lining. There is a silver lining between the limitations of PCD in the ROM and the aforementioned non-succinct construction of PCD in the ROM, which improves our understanding of security bounds in practice. Specifically, *the non-succinct construction of PCD in the ROM described above can be viewed as an idealization of succinct hash-based constructions of PCD in practice*, as we now explain.

The random oracle methodology tells us that the (non-relativized) SNARK $(\mathcal{P}_1, \mathcal{V}_1)$ in the ROM can be viewed as an idealization of a hash-based SNARK $(\hat{\mathcal{P}}_1, \hat{\mathcal{V}}_1)$ in the standard model, namely, the scheme $(\mathcal{P}_1, \mathcal{V}_1)$ where the random oracle is instantiated via a concrete hash function. Crucially, in a similar (though formally distinct) way, we can view $(\mathcal{P}_2, \mathcal{V}_2)$ as an idealization of $(\hat{\mathcal{P}}_1, \hat{\mathcal{V}}_1)$ when used to prove computations that involve calls to the concrete hash function. Indeed, $(\mathcal{P}_2, \mathcal{V}_2)$ equals $(\mathcal{P}_1, \mathcal{V}_1)$ up to the fact that calls to the random oracle are included as explicit input-output claims in the output argument string.

Next, let $(\hat{\mathbb{P}}, \hat{\mathbb{V}})$ be the hash-based PCD scheme in the standard model that is obtained by recursively composing the hash-based SNARK $(\hat{\mathcal{P}}_1, \hat{\mathcal{V}}_1)$. The PCD scheme $(\hat{\mathbb{P}}, \hat{\mathbb{V}})$ is the real-world hash-based construction whose concrete security we wish to understand. The key point in this discussion is that we can view the weak PCD scheme $(\mathbb{P}, \mathbb{V})$ in the ROM mentioned above as an idealization of $(\hat{\mathbb{P}}, \hat{\mathbb{V}})$. This is because:

- in the standard model, $(\hat{\mathbb{P}}, \hat{\mathbb{V}})$ is obtained via recursive composition of $(\hat{\mathcal{P}}_1, \hat{\mathcal{V}}_1)$;
- in the ROM, $(\mathbb{P}, \mathbb{V})$ is obtained via recursive composition of $(\mathcal{P}_2, \mathcal{V}_2)$;
- $(\mathcal{P}_2, \mathcal{V}_2)$ is an idealization of $(\hat{\mathcal{P}}_1, \hat{\mathcal{V}}_1)$ when used for computations involving calls to the hash function.

In sum, it may be reasonable to set the security parameters of $(\hat{\mathbb{P}}, \hat{\mathbb{V}})$ according to the security parameters of $(\mathbb{P}, \mathbb{V})$ (whose security is establish by our Theorem 1). This is tantamount to conjecturing that attacks against $(\hat{\mathbb{P}}, \hat{\mathbb{V}})$ are no more effective than attacks against $(\mathbb{P}, \mathbb{V})$. More precisely, either an attack against $(\hat{\mathbb{P}}, \hat{\mathbb{V}})$ reduces to an attack against $(\mathbb{P}, \mathbb{V})$ (inheriting its security), or an attack (usefully) exploits the instantiation (and non-black-box use of) the concrete hash function, which remains an open problem.

The above reasoning is summarized in Fig. 1 (and compared to prior approaches). This approach avoids the need to replace the random oracle with a hash function in the middle of the construction and its analysis; instead, all heuristics are deferred to the very end. (Deferring all heuristics to the very end has several advantages, articulated in [17]; indeed, other works achieving PCD in oracle models [16,21] also benefit from the ability to defer any heuristics till the end.)

Instantiating the SNARK. The aforementioned PCD construction in the ROM is based on a given SNARK in the ROM. There are several SNARKs in the ROM (with uncon-

ditional security) [6,23–25,36]. These constructions follow a common paradigm: they compile a probabilistic proof (a PCP or an IOP) into a SNARK by using a vector commitment scheme in the ROM and other ROM techniques. If the underlying probabilistic proof has a straightline extractor (the vast majority of relevant probabilistic proofs do) then the resulting SNARK in the ROM has one as well. Indeed, these constructions preserve straightline extractability (e.g., the vector commitment scheme in the ROM is straightline extractable).

For example, the SNARK in the ROM in [5] (known as *STARK*) is widely deployed in practice (along with various optimizations), including with recursion [43]. It is based on an IOP that is far more practical than any known PCP (and admits a straightline extractor). By our Theorem 1, the resulting PCD from the relativized version of the IOP-based SNARK has knowledge soundness error

$$\kappa_{\mathsf{PCD}}(\lambda, \mathsf{q}, \mathsf{s}, N) \leq \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}) \leq \mathsf{q} \cdot \kappa_{\mathsf{IOP}} + \frac{4\mathsf{q}^2}{2^\lambda} \quad ,$$

where κ_{IOP} is the (straightline) knowledge soundness error of the IOP. Note that in the ROM, κ_{ARG} does not depend on the adversary size.

Our bound provides partial justification for the security parameters for PCD based on this SNARK used in practice. (This was one of our initial motivations to study concrete security bounds for PCD in the ROM.)

Remark 2 (compatibility with zero knowledge). Recursively composing a zero-knowledge SNARK (that can prove correctness of its own verifier) yields zero-knowledge PCD. Concrete bounds on the zero-knowledge error of the PCD construction (in terms of the zero-knowledge error of the underlying SNARK) are known, including for the hash-based constructions of interest to us [20]. (Analyzing zero knowledge is "easy" because only the last recursion matters.)

Our proposal for a heuristic security analysis of the knowledge soundness error of hash-based PCD constructions is compatible with zero knowledge, in the following sense. Observe that if the (non-relativized) SNARK $(\mathcal{P}_1, \mathcal{V}_1)$ is zero knowledge, the transformation from $(\mathcal{P}_1, \mathcal{V}_1)$ to the relativized NARK $(\mathcal{P}_2, \mathcal{V}_2)$ does not necessarily maintain zero knowledge (due to the inclusion of query-answer pairs in the argument string). However, this is not a problem because our goal is to establish security bounds on the knowledge soundness error: the (non-succinct) PCD construction $(\mathbb{P}, \mathbb{V})$ obtained from $(\mathcal{P}_2, \mathcal{V}_2)$ remains an idealization of the hash-based PCD construction $(\hat{\mathbb{P}}, \hat{\mathbb{V}})$ that is obtained via recursive composition of $(\hat{\mathcal{P}}_1, \hat{\mathcal{V}}_1)$ (the hash-based instantiation of $(\mathcal{P}_1, \mathcal{V}_1)$ that heuristically remains zero knowledge). Hence, our proposal, in particular, also could be used as a guide for parameters of hash-based PCD constructions that are zero knowledge.

2.4 Example: A Real-World Compliance Predicate with Unknown Size and Depth Bound

We describe a compliance predicate with unbounded transcript size and depth that is an illustrative simplification of a compliance predicate deployed in a real-world application. Prior security analyses of PCD constructions do not provide any security guarantees (in terms of knowledge soundness) for such predicates. In contrast, the discussion

in Sect. 2.3 for the ROM (where, in particular, adversary size does not matter) explains how our Theorem 1 can be used to partially justify security parameters currently used in practice for this example. We elaborate on this below.

Motivation: Recursive STARKs. Computation in the Ethereum smart contract system is expensive: informally, each computation step is re-executed by every node in the network, and so the system charges users for each computation step that they want to execute (e.g., by calling a smart contract). A class of architectures known as *layer 2 proof-based rollups* [27] moves computation off-chain, in the sense that users send their computation requests to an aggregator who then periodically produces a SNARK proof about batches of user computations; the Ethereum smart contract system then verifies the SNARK proof and makes a state transition reflecting all the computations in the batch. The SNARK's succinctness property ensures that checking a SNARK proof is exponentially cheaper than checking the computation it attests to. These savings in on-chain computation are the motivation behind layer 2 proof-based rollups.

Producing SNARK proofs for large batches is expensive, but efficiency can be improved if the SNARK proof is itself produced via a PCD distributed computation that involves separately proving and aggregating small sub-computations, following a "proof tree" approach common in PCD applications [8,45]. This approach is taken by several systems, including one produced by StarkWare [30,43].

Informally, a smart contract on Ethereum [42] is a PCD verifier that enables the recursive proof composition of "STARK proofs" according to a compliance predicate described below.[11] The users submit computation requests by providing a piece of code to run and an input for it, which are the local data in the distributed computation. Messages, on the other hand, are hashes of outputs of computations.

In that system, security in the recursive composition of the STARK is assumed to equal the security of a standalone (non-recursive) use of the STARK (i.e., no security loss is accounted for in the security reduction from PCD to the STARK). Is this assumption (at least heuristically) justified?

The Compliance Predicate. As mentioned in Sect. 2.1, a compliance predicate receives as input, for a given vertex v in the graph (of the distributed computation), an output message z, some local data w_{loc}, and (in the recursive case) a list of input messages $(z_i)_i$. Let $h\colon \{0,1\}^* \to \{0,1\}^\lambda$ be a collision-resistant hash function, M be a universal Turing machine (on input a program P and an input x, M outputs $P(x)$), and $T \in \mathbb{N}$ be a maximum time bound. Below we describe a compliance predicate $\phi_{h,M,T}\colon \{0,1\}^* \to \{0,1\}$.

- *Formats:*
  - Local data w_{loc} is a tuple (P, x), where P is a program and x is an input.
  - A message z is a pair (y, t), where y is a claimed output (or hash value) and t is a time bound.
- *Base case: v is a source vertex.*
  $\phi_{h,M,T}(z, w_{\mathrm{loc}}, \bot)$:

[11] A STARK (as deployed in that system) is the heuristic instantiation (via the random oracle methodology) of a SNARK in the ROM, with straightline knowledge soundness, that is based on a certain IOP.

1. Parse z as (y, t).
2. Parse w_{loc} as (P, x).
3. Check that $t \leq T$, $M(P, x) = y$, and $M(P, x)$ runs in t steps.

– *Recursive case: v is an internal node.*

$\phi_{h,M,T}(z, w_{\text{loc}}, (z_i)_i)$:

1. Parse z as (y, t).
2. Check that $t = 0$ and $w_{\text{loc}} = \bot$.
3. For each i, parse z_i as (y_i, t_i) and check that $t_i \leq T$.
4. Check that $h((y_i)_i) = y$.

The base case in $\phi_{h,M,T}$ represents user computation requests, and the recursive case represents aggregation. In practice, h is set to a concrete hash function (e.g., blake2s in [42]), M is set to a specific universal machine (e.g., a machine that executes Cairo instructions [30]), and T to some large upper bound.

Moreover, $\phi_{h,M,T}$ does not impose any bound on the depth (or size) of a compliant PCD distributed computation: given $\lambda, T \in \mathbb{N}$, $\phi_{h,M,T}$ allows a chain of computations of any length, independent of λ and T, to be aggregated together. In particular, given a batch of base cases, it is possible to combine them in any arbitrary way by hashing the outputs of their computations for some unknown number of times.

In sum, *no prespecified upper bound N would support this compliance predicate*.

Sketch of the Application. We outline how the aggregator uses a PCD scheme $\mathsf{PCD} = (\mathbb{P}, \mathbb{V})$ relative to the compliance predicate $\phi_{h,M,T}$. Consider two computation requests (P_1, x_1, t_1) and (P_2, x_2, t_2), where P_i is a program, x_i is an input, and t_i is a time bound. The aggregator uses the PCD prover $\mathbb{P}$ to generate proofs π_1, π_2 attesting to the $\phi_{h,M,T}$-compliance of $(y_1, t_1), (y_2, t_2)$ respectively, where $y_1 := P(x_1), y_2 := P(x_2)$. Then, using these messages and proofs, the aggregator again uses the PCD $\mathbb{P}$ to create a proof Π attesting to the $\phi_{h,M,T}$-compliance of $(y, 0)$, where $y := h((y_1, y_2))$. More generally, the aggregator can generate a single compliance proof for a batch of computation requests $((P_i, x_i, t_i))_i$ as follows.

1. Compute the proofs for each user request (base case): For each i:
   – $y_i := M(P_i, x_i)$.
   – $\Pi_i \leftarrow \mathbb{P}(\phi_{h,M,T}, (y_i, t_i), (P_i, x_i), \bot)$
2. Compute the proofs for the second layer: For each neighboring pair (i, j) of the computation requests:
   – $\Pi_{i,j} \leftarrow \mathbb{P}(\phi_{h,M,T}, (h(y_i, y_j), 0), ((y_i, t_i), (y_j, t_j)), (\Pi_i, \Pi_j))$.
3. Keep aggregating proofs until there is only one node remaining. Output the last proof Π_{out}.

The above procedure always aggregates two proofs. In practice, users may submit computation requests in a streaming fashion. Hence, the above process can be generalized to handle streaming requests by greedily aggregating the requests submitted together (more than 2 requests can be handled at once). The resulting PCD transcript is not necessarily a binary tree.

This procedure is useful in multiple scenarios. One of them is when one or multiple users submit a series of computation requests: it is possible to construct *one* SNARK

proof for all the computations so that the Ethereum smart contract can verify this proof and update the states all together. More specifically, the aggregator divides all requests into smaller batches, and compliance predicate $\phi_{h,M,T}$ is able to handle each batch in parallel and combine them together into one proof even when the computations are not the same.

Prior work vs. our Result in this Application. Prior security analyses of PCD constructions (both from non-black-box extractors and from straightline extractors as in [16,21]) establish upper bounds on the PCD knowledge soundness error that depend on the transcript size and/or depth. Hence no security guarantees (for knowledge soundness) are provided for the compliance predicate $\phi_{h,M,T}$ deployed in [42] described above. Moreover, even if $\phi_{h,M,T}$ were modified to impose some pre-specified large transcript size/depth, the security loss depending on these parameters would have to be accounted for, which would cause a corresponding (and possibly large!) increase in the security parameters used in that system. (In other words, the underlying STARK would have to be much more secure to account for this loss.)

In contrast, our Theorem 1 establishes an upper bound on the PCD knowledge soundness error that applies to the compliance predicate $\phi_{h,M,T}$; indeed, in the ROM the adversary size does not matter, so the upper bound does not depend on either transcript size or depth (see the end of Sect. 1.1). In turn, since the underlying recursive STARKs are a (heuristic) PCD construction obtained from SNARKs in the ROM, our discussion in Sect. 2.3, provides new insights for practitioners. Specifically, that discussion suggests the security achieved by a non-recursive one-shot STARK proof is inherited by the corresponding PCD scheme without any loss. This provides a heuristic justification for the current settings of parameters in that system.

2.5 Technical Extension: A More General Analysis

The notion of straightline extraction for SNARKs that we used so far imposes two requirements on the knowledge extractor $\mathcal{E}$ (see Definition 1): (i) $\mathcal{E}$ is deterministic; and (ii) $\mathcal{E}$ does not query the oracle. These requirements are typically fulfilled by straightline extractors for known relativized SNARKs. Under these requirements Theorem 1 yields the upper bound $\kappa_{\mathsf{PCD}}(\lambda, \mathsf{q}, \mathsf{s}, N) \leq \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}, \mathsf{s} + O(N \cdot \mathsf{t}_{\mathsf{ARG}}(\lambda, \mathsf{q})))$.

We additionally ask: *how does the upper bound on κ_{PCD} change if we consider a notion of straightline extraction that relaxes either of these requirements?* Such relaxations can be useful; we give two examples.

- *Randomness.* The SNARK knowledge extractor $\mathcal{E}$ typically runs, as a subroutine, a knowledge extractor for an underlying probabilistic proof, which in turn may rely on a list-decoding algorithm for the error-correcting code used by the probabilistic proof. Randomness can be used to speed up list-decoding algorithms and, if $\mathcal{E}$ were required to be deterministic, those speedups would be ruled out.
- *Querying the oracle.* The SNARK knowledge extractor $\mathcal{E}$ may wish to query the oracle so to determine the decision bit of the SNARK verifier on the given instance and argument string (i.e., to compute $\mathcal{V}^f(\mathbb{x}, \pi)$).

In light of the above, we additionally give a general security analysis that upper bounds the PCD knowledge soundness error κ_{PCD} without assuming either of the requirements,

broadening the applicability of our result. While this analysis results in slightly larger bounds, the bounds remain significant improvements over the prior state of the art.

Below, we elaborate on how we handle each relaxation individually; in the technical sections, we handle both relaxations simultaneously (in which case the different types of upper bounds combine in a single upper bound for both). Fix the security parameter λ. The error κ_{ARG} primarily depends on the number of queries made by the adversary and the size of the adversary, which is the focus of the reasoning below. (The error κ_{ARG} depends also on other values; We refer the reader to the full version of this paper for all technical details.)

- *Probabilistic extractors.* The basic analysis described in Sect. 2.1 that leads to the upper bound $\kappa_{\mathsf{PCD}}(\lambda, \mathsf{q}, \mathsf{s}, N) \leq N \cdot \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}, \mathsf{s} + O(N \cdot \mathsf{t}_{\mathsf{ARG}}(\lambda, \mathsf{q})))$ can be adapted to hold for probabilistic extractors. For the i-th invocation of the SNARK extractor $\mathcal{E}$, we consider a corresponding malicious argument prover $\widetilde{\mathcal{P}}$ that outputs the message-proof pair in the label of the unique outgoing edge of the i-th vertex considered by $\mathbb{E}$. The extraction error for each malicious argument prover is upper bounded by $\kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}, \mathsf{s} + O(N \cdot \mathsf{t}_{\mathsf{ARG}}(\lambda, \mathsf{q})))$ (as the argument prover $\widetilde{\mathcal{P}}$ runs the q-query s-size PCD prover $\widetilde{\mathbb{P}}$ to obtain the output of the PCD transcript and then invokes the argument extractor for at most N times along with some post-processing). Finally, the PCD knowledge soundness error follows from a union bound.
  This bound is essentially tight. The knowledge soundness error of $\mathbb{E}$ may come either from the choice of oracle or from $\mathcal{E}$'s randomness. The latter case is something that, intuitively, must be paid N times, once per extraction; and it might be that, say, half of the knowledge soundness error is due to $\mathcal{E}$'s randomness. (Each invocation of $\mathcal{E}$ has an independent error from other invocations of $\mathcal{E}$, so the errors accumulate.)
  More generally, we could consider a definition of straightline knowledge soundness for the SNARK that separates a global error κ_{ARG} due to the oracle and a local error ϵ due to $\mathcal{E}$'s randomness. In this case, the upper bound would be $\kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}, \mathsf{s} + O(N \cdot \mathsf{t}_{\mathsf{ARG}}(\lambda, \mathsf{q}))) + N \cdot \epsilon$.
- *Extractors with oracle queries.* Suppose that the SNARK extractor $\mathcal{E}$ makes q' queries to the oracle. In the PCD extractor $\mathbb{E}$, we need to account for a query-answer trace that grows with each invocation of the SNARK extractor $\mathcal{E}$. Indeed, as per the definition of straightline knowledge soundness of the SNARK, each invocation of the SNARK extractor $\mathcal{E}$ takes in the query-answer trace of the corresponding malicious SNARK prover, which in the security reduction is an algorithm that runs the malicious PCD prover *plus* prior executions of $\mathcal{E}$ (each of which contributes new queries).
  A basic analysis here would establish a soundness error of roughly $\kappa_{\mathsf{PCD}}(\lambda, \mathsf{q}, \mathsf{s}, N) \leq \sum_{i=1}^{N} \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q} + i \cdot \mathsf{q}', \mathsf{s} + O(N \cdot \mathsf{t}_{\mathsf{ARG}}(\lambda, \mathsf{q})))$. Indeed, after i extractions, the query-answer trace for the execution of the i-th argument adversary, which initially has length at most q due to the malicious PCD prover, increases by at most $i \cdot \mathsf{q}'$. A union bound over all vertices gives the aforementioned bound.
  A more careful analysis establishes a (tight) bound of roughly $\kappa_{\mathsf{PCD}}(\lambda, \mathsf{q}, \mathsf{s}, N) \leq \sum_{i=1}^{N} \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q} + d_i \cdot \mathsf{q}', \mathsf{s} + O(N \cdot \mathsf{t}_{\mathsf{ARG}}(\lambda, \mathsf{q})))$, where d_i is the depth of the vertex associated with the i-th extraction in the extracted PCD transcript T (which can be exponentially smaller than size). Indeed, an extracted PCD transcript T is a tree;

when using the SNARK extractor $\mathcal{E}$ for a vertex v, the query-answer trace that "matters" for v only needs to include (the basic query-answer trace of the malicious PCD prover and) the queries and answers made by extractions *on the path from v to the root*. Hence, by giving each $\mathcal{E}$ only the query-answer traces it needs, the number of queries made by argument adversaries depends only on the depth of the PCD transcript.

3 Preliminaries

Definition 2. *An* **indexed relation** R *is a set of tuples* $(\mathbb{i}, \mathbb{x}, \mathbb{w})$ *where* $\mathbb{i}$ *is the index,* $\mathbb{x}$ *the instance, and* $\mathbb{w}$ *the witness. The corresponding* **indexed language** $L(R)$ *is the set of pairs* $(\mathbb{i}, \mathbb{x})$ *for which there exists a witness* $\mathbb{w}$ *such that* $(\mathbb{i}, \mathbb{x}, \mathbb{w}) \in R$.

Definition 3. *For a distribution over oracles $\mathcal{U}$, an* **oracle indexed relation** $R^{\mathcal{U}}$ *is a set of indexed relations* $\{R^f : f \in \mathcal{U}\}$.

Definition 4. *The* **query-answer trace** *of an algorithm A with oracle access to $f \in \mathcal{U}(\lambda)$ is a list* tr *of query-answer pairs that includes the queries made by A along with the corresponding answers by the oracle. We write $z \xleftarrow{\mathsf{tr}} A^f$ to mean that A, given oracle f, outputs z and has query-answer trace* tr.

3.1 Non-Interactive Arguments in Oracle Models

We provide notation and definitions for (preprocessing) non-interactive arguments as used in this paper. We do not describe the soundness property, as we always use a knowledge soundness property.

Definition 5. *A* **(preprocessing) non-interactive argument** *relative to an oracle distribution $\mathcal{U}$ for an oracle indexed relation $R^{\mathcal{U}}$ is a tuple of algorithms* $\mathsf{ARG} = (\mathcal{G}, \mathcal{I}, \mathcal{P}, \mathcal{V})$ *that works as follows.*

- $\mathcal{G}(1^\lambda) \to \mathsf{pp}$*: On input a security parameter λ (in unary), the generator $\mathcal{G}$ samples public parameters* pp.
- $\mathcal{I}^f(\mathsf{pp}, \mathbb{i}) \to (\mathsf{ipk}, \mathsf{ivk})$*: On input the public parameters* pp *and an index* $\mathbb{i}$ *for the relation R^f, the indexer $\mathcal{I}$ deterministically computes index-specific proving and verification keys* $(\mathsf{ipk}, \mathsf{ivk})$.
- $\mathcal{P}^f(\mathsf{ipk}, \mathbb{x}, \mathbb{w}) \to \pi$*: On input an index-specific proving key* ipk*, an instance* $\mathbb{x}$*, and a corresponding witness* $\mathbb{w}$*, the prover $\mathcal{P}$ computes an argument string π that attests to the claim that* $(\mathbb{i}, \mathbb{x}, \mathbb{w}) \in R^f$.
- $\mathcal{V}^f(\mathsf{ivk}, \mathbb{x}, \pi) \to b$*: On input an index-specific verification key* ivk*, and an instance* $\mathbb{x}$*, and a corresponding argument string π, the verifier $\mathcal{V}$ outputs a decision a bit b.*

Definition 6 (Perfect Completeness). *For every security parameter $\lambda \in \mathbb{N}$ and adversary $\mathcal{A}$,*

$$\Pr\left[\begin{array}{c|r} & f \leftarrow \mathcal{U}(\lambda) \\ (\mathbb{i}, \mathbb{x}, \mathbb{w}) \in R^f & \mathsf{pp} \leftarrow \mathcal{G}(1^\lambda) \\ \Downarrow & (\mathbb{i}, \mathbb{x}, \mathbb{w}) \leftarrow \mathcal{A}^f(\mathsf{pp}) \\ \mathcal{V}^f(\mathsf{ivk}, \mathbb{x}, \pi) = 1 & (\mathsf{ipk}, \mathsf{ivk}) \leftarrow \mathcal{I}^f(\mathsf{pp}, \mathbb{i}) \\ & \pi \leftarrow \mathcal{P}^f(\mathsf{ipk}, \mathbb{x}, \mathbb{w}) \end{array}\right] = 1 \ .$$

Definition 7 (Straightline Knowledge Soundness). ARG *has* **(straightline) knowledge soundness error** κ_{ARG} **with extraction time** $\mathsf{t}_{\mathsf{ARG}}$ *if there exists a* **deterministic** *extractor* $\mathcal{E}$ *such that, for every security parameter* $\lambda \in \mathbb{N}$*, auxiliary input distribution* $\mathcal{D}$*, query bound* $\mathsf{q}_{\widetilde{\mathcal{P}}} \in \mathbb{N}$*, size bound* $\mathsf{s}_{\widetilde{\mathcal{P}}} \in \mathbb{N}$*,* $\mathsf{q}_{\widetilde{\mathcal{P}}}$*-query* $\mathsf{s}_{\widetilde{\mathcal{P}}}$*-size deterministic circuit* $\widetilde{\mathcal{P}}$*, index size bound* $n \in \mathbb{N}$*, and instance size bound* $k \in \mathbb{N}$*,*

$$\Pr\left[\begin{array}{c|r} & f \leftarrow \mathcal{U}(\lambda) \\ |\mathbb{i}| \leq n & \mathsf{pp} \leftarrow \mathcal{G}(1^\lambda) \\ \wedge\ |\mathbb{x}| \leq k & \mathsf{ai} \leftarrow \mathcal{D}(\mathsf{pp}) \\ \wedge\ (\mathbb{i}, \mathbb{x}, \mathbb{w}) \notin R^f & (\mathbb{i}, \mathbb{x}, \pi) \xleftarrow{\mathsf{tr}} \widetilde{\mathcal{P}}^f(\mathsf{pp}, \mathsf{ai}) \\ \wedge\ \mathcal{V}^f(\mathsf{ivk}, \mathbb{x}, \pi) = 1 & (\mathsf{ipk}, \mathsf{ivk}) \leftarrow \mathcal{I}^f(\mathsf{pp}, \mathbb{i}) \\ & \mathbb{w} \leftarrow \mathcal{E}(\mathsf{pp}, \mathbb{i}, \mathbb{x}, \pi, \mathsf{tr}) \end{array}\right] \leq \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}_{\widetilde{\mathcal{P}}}, \mathsf{s}_{\widetilde{\mathcal{P}}}, n, k) \ ,$$

and $\mathcal{E}$ *runs in time* $\mathsf{t}_{\mathsf{ARG}}(\lambda, \mathsf{q}_{\widetilde{\mathcal{P}}}, n, k)$*.*

Remark 3. The auxiliary input distribution $\mathcal{D}$ can be the uniform random distribution. In that case the auxiliary input ai is a uniform random string, which enables the argument adversary $\widetilde{\mathcal{P}}$ to be randomized. In other words, $\widetilde{\mathcal{P}}$ is deterministic relative to the auxiliary input ai.

3.2 Proof-Carrying Data in Oracle Models

We provide notation and definitions for (preprocessing) proof-carrying data as used in this paper. This requires first introducing definitions for PCD transcripts and compliance.

Definition 8. *A* **(PCD) transcript** T *is a directed acyclic graph where each vertex* $u \in V(\mathrm{T})$ *is labeled by local data* $w_{\mathrm{loc}}^{(u)}$ *and each edge* $e \in E(\mathrm{T})$ *is labeled by a message* $z^{(e)} \neq \bot$*. The* **output** *of a transcript* T*, denoted* $\mathsf{out}(\mathrm{T})$*, is the message* $z^{(e)}$ *where* $e = (u, v)$ *is the lexicographically-first edge such that* v *is a sink.*

Definition 9. *A* **compliance predicate** ϕ *(with* M *input messages of size* l*) is an oracle boolean circuit that receives as input* 1 *output message of size at most* l*, some local data, and* M *input messages of size at most* l*, and outputs a decision bit. In particular,* ϕ *outputs* 0 *if more than* $M + 1$ *messages are given or any of the input messages are longer than* l *bits. We use* $|\phi|$ *to denote the size of the circuit* ϕ *and* $\mathsf{qnum}(\phi)$ *to denote the number of queries by* ϕ *to the oracle function.*

Definition 10. *Let* $\mathcal{U}$ *be an oracle distribution and let* Φ *be a class of compliance predicates. Consider* $f \in \mathcal{U}$ *and* $\phi \in \Phi$*. Given a transcript* T*, a vertex* $u \in V(\mathrm{T})$ *is* (ϕ, f)**-compliant** *if the following holds for every outgoing edge* $e = (u, v) \in E(\mathrm{T})$ *from* u*:*

- (base case) *if* u *has no incoming edges,* $\phi^f(z^{(e)}, w_{\mathrm{loc}}^{(u)}, (\bot)) = 1$*;*
- (recursive case) *if* u *has incoming edges* $e_1, \ldots, e_M$*,* $\phi^f(z^{(e)}, w_{\mathrm{loc}}^{(u)}, (z^{(e_1)}, \ldots, z^{(e_M)})) = 1$*.*

The transcript T *is* (ϕ, f)**-compliant** *if* $E(\mathrm{T}) \neq \emptyset$ *and every vertex* $u \in V(\mathrm{T})$ *is* (ϕ, f)*-compliant.*

Definition 11. *We define the depth, size, and arity of a PCD transcript* T.

- *The* **depth** $\mathsf{depth}(\mathrm{T})$ *is the number of vertices of the longest path in* T.
- *The* **size** $\mathsf{size}(\mathrm{T})$ *is the number of non-sink vertices in* T.
- *The* **arity** $\mathsf{arity}(\mathrm{T})$ *is the maximum number of incoming edges of any vertex in* T.

Definition 12. *We define the transcript depth, transcript size, and transcript arity of a compliance predicate* ϕ*. Let* $\mathcal{U}$ *be an oracle distribution.*

- *The* **transcript depth** *is*

$$\mathsf{tdepth}(\phi) \coloneqq \max_{\substack{f \in \mathcal{U} \\ \mathrm{T}\ \textit{is}\ (\phi,f)\textit{-compliant}}} \mathsf{depth}(\mathrm{T}) \ ;$$

- *The* **transcript size** *is*

$$\mathsf{tsize}(\phi) \coloneqq \max_{\substack{f \in \mathcal{U} \\ \mathrm{T}\ \textit{is}\ (\phi,f)\textit{-compliant}}} \mathsf{size}(\mathrm{T}) \ ;$$

- *The* **transcript arity** *is*

$$\mathsf{tarity}(\phi) \coloneqq \max_{\substack{f \in \mathcal{U} \\ \mathrm{T}\ \textit{is}\ (\phi,f)\textit{-compliant}}} \mathsf{arity}(\mathrm{T}) \ .$$

Definition 13. *A compliance predicate* ϕ *is* (Φ, N, D, M, S, Q)**-compatible** *if:* $\phi \in \Phi$*;* $\mathsf{tsize}(\phi) \leq N$*;* $\mathsf{tdepth}(\phi) \leq D$*;* $\mathsf{tarity}(\phi) \leq M$*;* $|\phi| \leq S$*; and* $\mathsf{qnum}(\phi) \leq Q$.

Definition 14. *A* **proof-carrying data scheme (PCD scheme)** *for a class of compliance predicates* Φ *relative to an oracle distribution* $\mathcal{U}$ *is a tuple of algorithms* $\mathsf{PCD} = (\mathbb{G}, \mathbb{I}, \mathbb{P}, \mathbb{V})$ *that works as follows.*

- $\mathbb{G}(1^\lambda) \to \mathbb{pp}$*: On input a security parameter* λ *(in unary), the generator* $\mathbb{G}$ *samples public parameters* $\mathbb{pp}$.
- $\mathbb{I}^f(\mathbb{pp}, \phi) \to (\mathbb{ipk}, \mathbb{ivk})$*: On input the public parameters* $\mathbb{pp}$ *and the compliance predicate* ϕ*, the indexer* $\mathbb{I}$ *deterministically computes proving and verification keys* $(\mathbb{ipk}, \mathbb{ivk})$.
- $\mathbb{P}^f(\mathbb{ipk}, z, w_{\mathrm{loc}}, ((z_i, \boldsymbol{\Pi}_i))_{i \in [M]}) \to \boldsymbol{\Pi}$ *: On input the proving key* $\mathbb{ipk}$*, a message* z*, a local data* w_{loc}*, and a list of incoming messages and proofs* $((z_i, \boldsymbol{\Pi}_i))_{i \in [M]}$ *, the prover* $\mathbb{P}$ *outputs a new proof* $\boldsymbol{\Pi}$ *for the outgoing message* z.
- $\mathbb{V}^f(\mathbb{ivk}, z, \boldsymbol{\Pi}) \to b$ *: On input the verification key* $\mathbb{ivk}$*, a message* z*, and a corresponding proof* $\boldsymbol{\Pi}$ *, the verifier* $\mathbb{V}$ *computes a decision bit* b.

Definition 15 (Perfect Completeness). *For every security parameter $\lambda \in \mathbb{N}$ and adversary $\mathbb{A}$,*

$$\Pr\left[\begin{array}{c}\left(\begin{array}{c}\phi \in \Phi \\ \wedge \left((\wedge_{i=1}^{M} z_i = \bot) \vee (\wedge_{i=1}^{M} \mathbb{V}^f(\mathbb{ivk}, z_i, \mathbb{\Pi}_i) = 1)\right) \\ \wedge \phi^f(z, w_{\mathrm{loc}}, (z_1, \ldots, z_M)) = 1 \\ \Downarrow \\ \mathbb{V}^f(\mathbb{ivk}, z, \mathbb{\Pi}) = 1\end{array}\right) \end{array} \middle| \begin{array}{r} f \leftarrow \mathcal{U}(\lambda) \\ \mathbb{pp} \leftarrow \mathbb{G}(1^\lambda) \\ (\phi, z, w_{\mathrm{loc}}, ((z_i, \mathbb{\Pi}_i))_{i=1}^{M}) \leftarrow \mathbb{A}^f(\mathbb{pp}) \\ (\mathbb{ipk}, \mathbb{ivk}) \leftarrow \mathbb{I}^f(\mathbb{pp}, \phi) \\ \mathbb{\Pi} \leftarrow \mathbb{P}^f(\mathbb{ipk}, z, w_{\mathrm{loc}}, ((z_i, \mathbb{\Pi}_i))_{i=1}^{M}) \end{array}\right] = 1 \ .$$

Definition 16 (Straightline Knowledge Soundness). PCD *has* **(straightline) knowledge soundness error** κ_{PCD} **with extraction time** $\mathsf{t}_{\mathsf{PCD}}$ *if there exists a* **deterministic** *extractor $\mathbb{E}$ such that, for every security parameter $\lambda \in \mathbb{N}$, auxiliary input distribution $\mathcal{D}$, indexer query bound $\mathsf{q}_{\mathbb{I}} \in \mathbb{N}$, indexer size bound $s_{\mathbb{I}}$, adversary query bound $\mathsf{q}_{\widetilde{\mathbb{P}}} \in \mathbb{N}$, adversary size bound $s_{\widetilde{\mathbb{P}}} \in \mathbb{N}$, $\mathsf{q}_{\widetilde{\mathbb{P}}}$-query $s_{\widetilde{\mathbb{P}}}$-size deterministic circuit $\widetilde{\mathbb{P}}$, predicate size bound $N \in \mathbb{N}$, predicate depth bound $D \in \mathbb{N}$, predicate circuit size bound $S \in \mathbb{N}$, predicate query number bound $Q \in \mathbb{N}$, number of input edges bound $M \in \mathbb{N}$, and message size bound $l \in \mathbb{N}$,*

$$\Pr\left[\begin{array}{c}\phi \text{ is } (\Phi, N, D, M, S, Q)\text{-compatible} \\ \wedge\ |z_{\mathrm{out}}| \leq l \\ \wedge\ \mathbb{V}^f(\mathbb{ivk}, z_{\mathrm{out}}, \mathbb{\Pi}_{\mathrm{out}}) = 1 \\ \wedge\ (\mathrm{T} \text{ is not } (\phi, f)\text{-compliant} \vee \mathsf{out}(\mathrm{T}) \neq z_{\mathrm{out}})\end{array} \middle| \begin{array}{r} f \leftarrow \mathcal{U}(\lambda) \\ \mathbb{pp} \leftarrow \mathbb{G}(1^\lambda) \\ \mathsf{ai} \leftarrow \mathcal{D}(\mathbb{pp}) \\ (\phi, z_{\mathrm{out}}, \mathbb{\Pi}_{\mathrm{out}}) \xleftarrow{\mathsf{tr}} \widetilde{\mathbb{P}}^f(\mathbb{pp}, \mathsf{ai}) \\ (\mathbb{ipk}, \mathbb{ivk}) \leftarrow \mathbb{I}^f(\mathbb{pp}, \phi) \\ \mathrm{T} \leftarrow \mathbb{E}(\mathbb{pp}, \mathbb{ivk}, \phi, z_{\mathrm{out}}, \mathbb{\Pi}_{\mathrm{out}}, \mathsf{tr}) \end{array}\right]$$

$$\leq \kappa_{\mathsf{PCD}}(\lambda, \mathsf{q}_{\mathbb{I}}, s_{\mathbb{I}}, \mathsf{q}_{\widetilde{\mathbb{P}}}, s_{\widetilde{\mathbb{P}}}, N, D, S, Q, M, l) \ ,$$

and $\mathbb{E}$ runs in time $\mathsf{t}_{\mathsf{PCD}}(\lambda, \mathsf{q}_{\mathbb{I}}, s_{\mathbb{I}}, \mathsf{q}_{\widetilde{\mathbb{P}}}, N, D, S, Q, M, l)$.

4 From Relativized ARG to PCD: Construction

We describe how to construct a PCD scheme from a relativized non-interactive argument (Theorem 3), and after that we describe how to construct a straightline PCD extractor from an underlying straightline non-interactive argument extractor (Theorem 4).

These constructions are straightforward adaptations to the relativized case of prior constructions in the literature ([7,8,20,21]). Our main contribution is the security analysis of the PCD scheme (via this straightline PCD extractor), which we postpone to Sect. 5.

Let $\mathcal{U}$ be an oracle distribution. The definition below is a circuit used to realize the recursive composition.

Construction 2. *Fix $\lambda \in \mathbb{N}$. Let $f \in \mathcal{U}(\lambda)$. Let $\mathcal{V}^{(\lambda,n,k)}$ be the circuit corresponding to the ARG verifier $\mathcal{V}$ with security parameter λ, checking indices of sizes at most n and instances of size at most k.*

$[C_{\mathcal{V},\phi}^{(\lambda,M,n,k)}]^f((\mathsf{ivk}, z_{\mathrm{out}}), (w_{\mathrm{loc}}, \boldsymbol{z}_{\mathrm{in}}, \boldsymbol{\pi}_{\mathrm{in}}))$*:*

1. *Check that $\phi^f(z_{\mathrm{out}}, w_{\mathrm{loc}}, \boldsymbol{z}_{\mathrm{in}}) = 1$.*
2. *If there exists i such that $(\boldsymbol{z}_{\mathrm{in}}[i], \boldsymbol{\pi}_{\mathrm{in}}[i]) \neq \bot$: check that $[\mathcal{V}^{(\lambda,n,k)}]^f(\mathsf{ivk}, (\mathsf{ivk}, \boldsymbol{z}_{\mathrm{in}}[i]), \boldsymbol{\pi}_{\mathrm{in}}[i]) = 1$ for every $i \in [M]$.*

Construction 3 (PCD from ARG). *Let* $\mathsf{ARG} = (\mathcal{I}, \mathcal{P}, \mathcal{V})$ *be a non-interactive argument for the oracle indexed relation* $R^{\mathcal{U}}_{\mathrm{CSAT}}$. *We construct a PCD scheme* $\mathsf{PCD} = (\mathbb{G}, \mathbb{I}, \mathbb{P}, \mathbb{V})$ *as follows.*

- $\mathbb{G}(1^\lambda)$*:*
  1. *Sample public parameters* $\mathsf{pp} \leftarrow \mathcal{G}(1^\lambda)$.
  2. *Output* $\mathbb{pp} := \mathsf{pp}$.
- $\mathbb{I}^f(\mathbb{pp}, \phi)$*:*
  1. *Parse* $\mathbb{pp}$ *as* pp.
  2. *Construct the oracle recursion circuit* $C := C^{(\lambda,M,n,k)}_{\mathcal{V},\phi}$.
  3. *Compute the index key pair* $(\mathsf{ipk}, \mathsf{ivk}) \leftarrow \mathcal{I}^f(\mathsf{pp}, C)$.
  4. *Output* $(\mathbb{ipk}, \mathbb{ivk}) := ((\mathsf{ipk}, \mathsf{ivk}), \mathsf{ivk})$.
- $\mathbb{P}^f(\mathbb{ipk}, z_{\mathrm{out}}, (w_{\mathrm{loc}}, \boldsymbol{z}_{\mathrm{in}}, \mathbb{\Pi}_{\mathrm{in}}))$*:*
  1. *Parse the proving key* $\mathbb{ipk}$ *as* $(\mathsf{ipk}, \mathsf{ivk})$.
  2. $\pi_{\mathrm{out}} \leftarrow \mathcal{P}^f(\mathsf{ipk}, (\mathsf{ivk}, z_{\mathrm{out}}), (w_{\mathrm{loc}}, \boldsymbol{z}_{\mathrm{in}}, \boldsymbol{\pi}_{\mathrm{in}}))$.
  3. *Output* π_{out}.
- $\mathbb{V}^f(\mathbb{ivk}, z_{\mathrm{out}}, \mathbb{\Pi}_{\mathrm{out}})$*:*
  1. *Parse the verification key* $\mathbb{ivk}$ *as* ivk.
  2. *Parse the PCD proof* $\mathbb{\Pi}_{\mathrm{out}}$ *as an argument proof* π_{out}.
  3. *Check that* $\mathcal{V}^f(\mathsf{ivk}, (\mathsf{ivk}, z_{\mathrm{out}}), \pi_{\mathrm{out}}) = 1$.

Construction 4 (Knowledge extractor for PCD). *Let* $\mathcal{E}$ *be a straightline knowledge extractor for* ARG. *We construct a straightline knowledge extractor* $\mathbb{E}$ *for* PCD *as follows.*

$\mathbb{E}(\mathbb{pp}, \mathbb{ivk}, \phi, z_{\mathrm{out}}, \mathbb{\Pi}_{\mathrm{out}}, \mathsf{tr})$*:*

1. *Parse* $\mathbb{pp}$ *as* pp.
2. *Parse* $\mathbb{\Pi}_{\mathrm{out}}$ *as* π_{out}.
3. *Parse* $\mathbb{ivk}$ *as* ivk.
4. *Initialize graph* $\mathrm{T} = (V, E)$ *where* $V = \{v_0, v_1\}$ *and* $E = \{(v_1, v_0)\}$.
5. *Label the edge* (v_1, v_0) *by* $(z_{\mathrm{out}}, \pi_{\mathrm{out}})$.
6. *Initialize the extraction queue* $\mathcal{L} := (v_1)$.
7. *Set* $\mathbb{i} := C^{(\lambda,M,n,k)}_{\mathcal{V},\phi}$.
8. *While* $\mathcal{L}$ *is non-empty:*
   (a) *Let* v *be the first vertex in* $\mathcal{L}$*, remove the first vertex* v *from* $\mathcal{L}$.
   (b) *Let* $z^{(e)}$ *and* $\pi^{(e)}$ *be the message and proof in the label of the unique outgoing edge* e *from* v.
   (c) *Let* $(\mathbb{i}_v, \mathbb{x}_v, \pi_v) := (\mathbb{i}, (\mathsf{ivk}, z^{(e)}), \pi^{(e)})$.
   (d) *Run the argument extractor* $\mathbb{w}_v \leftarrow \mathcal{E}(\mathsf{pp}, \mathbb{i}_v, \mathbb{x}_v, \pi_v, \mathsf{tr})$.
   (e) *Parse* $(w^{(v)}_{\mathrm{loc}}, \boldsymbol{z}^{(v)}_{\mathrm{in}}, \mathbb{\Pi}^{(v)}_{\mathrm{in}})$.
   (f) *Label* v *in* T *by* $w^{(v)}_{\mathrm{loc}}$.
   (g) *For every* j *such that* $\boldsymbol{z}^{(v)}_{\mathrm{in}}[j] \neq \bot$*:*
      i. *Add a new vertex* v' *to* V*, and an edge* (v', v) *in* E.
      ii. *Parse* $\mathbb{\Pi}^{(v)}_{\mathrm{in}}[j]$ *as* $\pi^{(v',v)}$.
      iii. *Add the label* $(\boldsymbol{z}^{(v)}_{\mathrm{in}}[j], \pi^{(v',v)})$ *to the edge* (v', v) *in* T.
      iv. *Add* v' *to* $\mathcal{L}$.
9. *Output the augmented transcript* T.

5 From Relativized ARG to PCD: Security Reduction

In Sect. 4, we described how to construct a PCD scheme from a relativized non-interactive argument, and how to construct a straightline PCD extractor from an underlying straightline non-interactive argument extractor. In this section, we give our security analysis of the PCD scheme, via this straightline PCD extractor.

Let $\mathcal{U}$ be an oracle distribution. Suppose that $\mathsf{ARG} = (\mathcal{G}, \mathcal{I}, \mathcal{P}, \mathcal{V})$ is a non-interactive argument for the oracle CSAT relation $R_{\mathrm{CSAT}}^{\mathcal{U}}$ with straightline knowledge soundness error $\kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}_{\tilde{\mathcal{P}}}, \mathsf{s}_{\tilde{\mathcal{P}}}, n, k)$ and extraction time $\mathsf{t}_{\mathsf{ARG}}(\lambda, \mathsf{q}_{\tilde{\mathcal{P}}}, n, k)$. Recall that $\lambda \in \mathbb{N}$ denotes the security parameter, $S \in \mathbb{N}$ the bound on the predicate circuit size, $M \in \mathbb{N}$ the bound on the arity of the PCD transcript, and l the bound on the size of each message. We define an index size bound n and instance size bound k:

- $n := \mathsf{nsize}(\lambda, S, M, l)$, where $\mathsf{nsize}(\cdot)$ is carefully defined in Lemma 1; and
- $k := |\mathsf{ivk}| + l$.

Theorem 5. *The PCD scheme* $\mathsf{PCD} = (\mathbb{G}, \mathbb{I}, \mathbb{P}, \mathbb{V})$ *constructed from* ARG *using Theorem 3 has straightline knowledge soundness error* $\kappa_{\mathsf{PCD}} = \kappa_{\mathsf{PCD}}(\lambda, \mathsf{q}_{\mathbb{I}}, s_{\mathbb{I}}, \mathsf{q}_{\tilde{\mathbb{P}}}, s_{\tilde{\mathbb{P}}}, N, D, S, Q, M, l)$ *and extraction time* $\mathsf{t}_{\mathsf{PCD}} = \mathsf{t}_{\mathsf{PCD}}(\lambda, \mathsf{q}_{\mathbb{I}}, s_{\mathbb{I}}, \mathsf{q}_{\tilde{\mathbb{P}}}, N, D, S, Q, M, l)$ *such that*

$$\kappa_{\mathsf{PCD}} \leq \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}_{\tilde{\mathcal{P}}}, \mathsf{s}_{\tilde{\mathcal{P}}}, n, k) \ ,$$
$$\mathsf{t}_{\mathsf{PCD}} \leq N \cdot (\mathsf{poly}(\log N, \log M, l, \mathsf{arglen}(n, k)) + \mathsf{t}_{\mathsf{ARG}}(\lambda, \mathsf{q}_{\tilde{\mathbb{P}}} + \mathsf{q}_{\mathbb{I}}, n, k)) \ ,$$

where

- $\mathsf{q}_{\tilde{\mathcal{P}}} := \mathsf{q}_{\tilde{\mathbb{P}}} + \mathsf{q}_{\mathbb{I}} + N \cdot Q$;
- $\mathsf{s}_{\tilde{\mathcal{P}}} := s_{\tilde{\mathbb{P}}} + s_{\mathbb{I}} + N \cdot S + N \cdot \mathsf{vsize}(\lambda, n, k) + \mathsf{t}_{\mathsf{PCD}}$, *where* $\mathsf{vsize}(\lambda, n, k)$ *is the size of the argument verifier* $\mathcal{V}$ *when invoked with security parameter* λ*, index of size* n*, and instance of size* k*.*
- $\mathsf{arglen}(\lambda, n, k)$ *is the size of the argument proof* π *outputted by the argument prover when invoked with security parameter* λ*, index of size* n*, and instance of size* k*.*

We analyze the knowledge soundness error in Sect. 5.1 and the extraction time in Sect. 5.2.

Remark 4 (The Information-Theoretic Setting). We discuss a special case of Theorem 5 that yields an even stronger result in a notable setting. Suppose that ARG has straightline knowledge soundness error κ_{ARG} that does not depend on the size of the adversary $\mathsf{s}_{\tilde{\mathcal{P}}}$; for example, this is the case in the "pure" random oracle setting, where adversaries may be computationally unbounded (and are limited only in the number of queries to the random oracle). Suppose further that the compliance predicate ϕ does not query the oracle ($Q = 0$); this is a common case as typically the oracle appears only in the argument verifier (not part of the compliance predicate). In this case the knowledge error is as follows:

$$\kappa_{\mathsf{PCD}} \leq \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}_{\tilde{\mathbb{P}}} + \mathsf{q}_{\mathbb{I}}, n, k) \ .$$

5.1 Knowledge Soundness Error

The analysis below is stated for non-interactive arguments with split verification (Definition 17), a property that holds essentially without loss of generality (Remark 5).

Definition 17. $\mathsf{ARG} = (\mathcal{G}, \mathcal{I}, \mathcal{P}, \mathcal{V})$ *has* **split verification** *if the argument string contains a list of query-answer pairs* tr *that the verifier* $\mathcal{V}$ *checks during verification. More precisely,* $\mathcal{V}$ *can be written as follows:*

$\mathcal{V}^f(\mathsf{ivk}, \mathbb{x}, \pi)$*:*
1. *Check that* $\mathsf{VerifyProof}(\mathsf{ivk}, \mathbb{x}, \pi) = 1$.
2. *Check that* $\mathsf{VerifyTrace}^f(\pi) = 1$.

The above subroutines are defined as follows:

- $\mathsf{VerifyProof}(\mathsf{ivk}, \mathbb{x}, \pi)$*: Parse* π *as* (π', tr) *and check that* $\mathcal{V}^{\mathsf{tr}}(\mathsf{ivk}, \mathbb{x}, \pi') = 1$. *(Output* 0 *if* $\mathcal{V}$ *makes a query* q *that is not contained in* tr.
- $\mathsf{VerifyTrace}^f(\pi)$*: Parse* π *as* (π', tr)*, and check that, for every* $(q, a) \in \mathsf{tr}$*,* $f(q) = a$.

Remark 5. Any non-interactive argument system can be modified to satisfy Definition 17, by augmenting the argument string with the list of query-answer pairs to be made by the argument verifier. We assume split verification throughout this proof.

Let $\widetilde{\mathbb{P}}$ be a $\mathsf{q}_{\widetilde{\mathbb{P}}}$-query PCD prover. Our goal is to upper bound the following expression:

$$\Pr\left[\begin{array}{c} \phi \text{ is } (\Phi, N, D, M, S, Q)\text{-compatible} \\ \wedge\ |z_{\text{out}}| \le l \\ \wedge\ \mathbb{V}^f(\mathbb{ivk}, z_{\text{out}}, \mathbb{\Pi}_{\text{out}}) = 1 \\ \wedge\ (\mathrm{T} \text{ is not } (\phi, f)\text{-compliant} \vee \mathsf{out}(\mathrm{T}) \neq z_{\text{out}}) \end{array} \middle| \begin{array}{r} f \leftarrow \mathcal{U}(\lambda) \\ \mathbb{pp} \leftarrow \mathbb{G}(1^\lambda) \\ \mathsf{ai} \leftarrow \mathcal{D}(\mathbb{pp}) \\ (\phi, z_{\text{out}}, \mathbb{\Pi}_{\text{out}}) \xleftarrow{\mathsf{tr}} \widetilde{\mathbb{P}}^f(\mathbb{pp}, \mathsf{ai}) \\ (\mathbb{ipk}, \mathbb{ivk}) \leftarrow \mathbb{I}^f(\mathbb{pp}, \phi) \\ \mathrm{T} \leftarrow \mathbb{E}(\mathbb{pp}, \mathbb{ivk}, \phi, z_{\text{out}}, \mathbb{\Pi}_{\text{out}}, \mathsf{tr}) \end{array}\right]. \tag{1}$$

As explained in Sect. 2.1, we construct an argument prover $\widetilde{\mathcal{P}}$.

$\widetilde{\mathcal{P}}^f(\mathsf{pp}, \mathsf{ai})$:
1. Set $\mathbb{pp} := \mathsf{pp}$.
2. Run $(\phi, z_{\text{out}}, \mathbb{\Pi}_{\text{out}}) \xleftarrow{\mathsf{tr}_{\widetilde{\mathbb{P}}}} \widetilde{\mathbb{P}}^f(\mathbb{pp}, \mathsf{ai})$.
3. Run $(\mathbb{ipk}, \mathbb{ivk}) \xleftarrow{\mathsf{tr}_{\mathbb{I}}} \mathbb{I}^f(\phi)$.
4. Set $\mathsf{tr} := \mathsf{tr}_{\widetilde{\mathbb{P}}} \,\|\, \mathsf{tr}_{\mathbb{I}}$.
5. Parse the PCD proof $\mathbb{\Pi}_{\text{out}}$ as an argument string π_{out}.
6. Parse the PCD verification key $\mathbb{ivk}$ as an argument verification key ivk.
7. Initialize a graph $\mathrm{T} = (V, E)$ where $V = \{v_0, v_1\}$ and $E = \{(v_1, v_0)\}$.
8. Label the edge (v_1, v_0) by $(z_{\text{out}}, \pi_{\text{out}})$.
9. Initialize the extraction queue as $\mathcal{L} := (v_1)$.
10. Set $\mathbb{i} := C_{\mathcal{V},\phi}^{(\lambda, M, n, k)}$.
11. While the extraction queue $\mathcal{L}$ is non-empty:
    (a) Let v be the first vertex in $\mathcal{L}$, remove the first vertex v from $\mathcal{L}$.

(b) Let $z^{(e)}$ and $\pi^{(e)}$ be the message and proof in the label of the unique outgoing edge e from v.
(c) Let $(\mathbb{i}_v, \mathbb{x}_v, \pi_v) := (\mathbb{i}, (\mathsf{ivk}, z^{(e)}), \pi^{(e)})$.
(d) Run the argument extractor $\mathbb{w}_v \leftarrow \mathcal{E}(\mathsf{pp}, \mathbb{i}_v, \mathbb{x}_v, \pi_v, \mathsf{tr})$.
(e) Parse $\mathbb{w}_v$ as $(w_{\mathrm{loc}}^{(v)}, \boldsymbol{z}_{\mathrm{in}}^{(v)}, \boldsymbol{\Pi}_{\mathrm{in}}^{(v)})$.
(f) Label the vertex v in T by $w_{\mathrm{loc}}^{(v)}$.
(g) For every j such that $\boldsymbol{z}_{\mathrm{in}}^{(v)}[j] \neq \bot$:
  i. Add a new vertex v' to V, and an edge (v', v) in E.
  ii. Parse $\boldsymbol{\Pi}_{\mathrm{in}}^{(v)}[j]$ as $\pi^{(v',v)}$.
  iii. Add the label $(\boldsymbol{z}_{\mathrm{in}}^{(v)}[j], \pi^{(v',v)})$ to the edge (v', v) in T.
  iv. Add the vertex v' to the extraction queue $\mathcal{L}$.
(h) Let $(v_1, \ldots, v_M)$ be the child-vertices just added for v (maybe there are less than M child-vertices of v in T, but the exact number does not matter as long as it is upper-bounded by M).
(i) Let e be the outgoing edge of v, check if at least one of the following is true:
  i. $\phi^f(z^{(e)}, w_{\mathrm{loc}}^{(v)}, (z^{(v_1,v)}, \ldots, z^{(v_M,v)})) \neq 1$.
  ii. There exists $i \in [M]$ such that $\mathsf{VerifyProof}(\mathsf{ivk}, (\mathsf{ivk}, z^{(v_i,v)}), \pi^{(v_i,v)}) \neq 1$, where $\mathsf{VerifyProof}$ is as defined in Definition 17.
  If the check passes, then output $(\mathbb{i}, (\mathsf{ivk}, z^{(e)}), \pi^{(e)})$.
12. Output $(\mathbb{i}, (\mathsf{ivk}, z^{(e)}), \pi^{(e)})$ where e is the topologically first edge in E. (This is a default output.)

The argument prover $\widetilde{\mathcal{P}}$ queries f when $\widetilde{\mathbb{P}}$, $\mathbb{I}$, and (each time it runs) ϕ. Hence the query complexity of $\widetilde{\mathcal{P}}$ can be upper bounded as follows:

$$\mathsf{q}_{\widetilde{\mathcal{P}}} \leq \mathsf{q}_{\widetilde{\mathbb{P}}} + \mathsf{q}_{\mathbb{I}} + N \cdot Q \ .$$

Similarly, the size of $\widetilde{\mathcal{P}}$ can be upper bounded as follows:

$$\mathsf{s}_{\widetilde{\mathcal{P}}} \leq s_{\widetilde{\mathbb{P}}} + s_{\mathbb{I}} + N \cdot S + N \cdot \mathsf{vsize}(\lambda, n, |\mathsf{ivk}| + l) + \mathsf{t}_{\mathsf{PCD}} \ .$$

For the rest of the discussion, we consider the following experiment:

Experiment 6

$$\left[\begin{array}{r} f \leftarrow \mathcal{U}(\lambda) \\ \mathsf{pp} \leftarrow \mathcal{G}(1^\lambda) \\ \mathbb{pp} := \mathsf{pp} \\ \mathsf{ai} \leftarrow \mathcal{D}(\mathsf{pp}) \\ (\phi, z_{\mathrm{out}}, \boldsymbol{\Pi}_{\mathrm{out}}) \xleftarrow{\mathsf{tr}_{\widetilde{\mathbb{P}}}} \widetilde{\mathbb{P}}^f(\mathbb{pp}, \mathsf{ai}) \\ (\mathbb{ipk}, \mathbb{ivk}) \leftarrow \mathbb{I}^f(\phi) \\ \mathrm{T} \leftarrow \mathbb{E}(\mathbb{pp}, \mathbb{ivk}, \phi, z_{\mathrm{out}}, \boldsymbol{\Pi}_{\mathrm{out}}, \mathsf{tr}_{\widetilde{\mathbb{P}}}) \\ (\mathbb{i}, \mathbb{x}, \pi) \xleftarrow{\mathsf{tr}_{\widetilde{\mathcal{P}}}} \widetilde{\mathcal{P}}^f(\mathsf{pp}, \mathsf{ai}) \\ (\mathsf{ivk}, \mathsf{ipk}) \leftarrow \mathcal{I}(\mathsf{pp}, \mathbb{i}) \\ \mathbb{w} \leftarrow \mathcal{E}(\mathsf{pp}, \mathbb{i}, \mathbb{x}, \pi, \mathsf{tr}_{\widetilde{\mathcal{P}}}) \end{array}\right]$$

To bound the probability in Eq. (1), we note that the condition $\mathsf{out}(\mathrm{T}) = z_{\mathrm{out}}$ always holds by Theorem 4, so we focus on the probability that T is not (ϕ, f)-compliant. Intuitively, our goal is to reduce the probability of T being not (ϕ, f)-compliant to the probability that the argument prover $\widetilde{\mathcal{P}}$ constructed above successfully outputs an argument string that fools the argument verifier.

Towards this, we use a notion called *strong* (ϕ, f)*-compliance*, which requires all vertices to be (ϕ, f)-compliant and also every index-instance-proof tuple associated with an edge to be accepted by the argument verifier. If a transcript T is strongly (ϕ, f)-compliant then, in particular, it is (ϕ, f)-compliant.

Definition 18. *A transcript* $\mathrm{T} = (V, E)$ *is* **strongly** (ϕ, f)**-compliant** *if the following holds:*

- *For every vertex* $u \in V$*,* u *is* (ϕ, f)*-compliant.*
- *For every edge* $e \in E$*,* $\mathcal{V}^f(\mathsf{ivk}, (\mathsf{ivk}, z^{(e)}), \pi^{(e)}) = 1$.

Remark 6. The transcript T output by the PCD extractor $\mathbb{E}$ in Theorem 4 is a tree, so every vertex $v \in V(\mathrm{T})$ has a unique outgoing edge. Definitions 10 and 18 can be simplified accordingly for the purpose of this discussion.

If the transcript T in Theorem 6 is not (ϕ, f)-compliant, then it must be the case that T is not strongly (ϕ, f)-compliant, which implies that there exists $v \in V(\mathrm{T})$ such that at least one of the following is true:

1. $\phi^f(z^{(e)}, w_{\mathrm{loc}}^{(v)}, (z^{(v_1,v)}, \ldots, z^{(v_M,v)})) \neq 1$,
2. $\mathcal{V}^f(\mathsf{ivk}, (\mathsf{ivk}, z^{(v_i,v)}), \pi^{(v_i,v)}) \neq 1$ for some $i \in [M]$,

where $e = (v, v')$ is the unique outgoing edge of v and $v_1, \ldots, v_M$ are child-vertices of v. Hence, by definition of $C_{\mathcal{V},\phi}^{(\lambda,M,n,k)}$ (Theorem 2), $(\mathbb{i}_v, \mathbb{x}_v, \mathbb{w}_v) := (C_{\mathcal{V},\phi}^{(\lambda,M,n,k)}, (\mathsf{ivk}, z^{(e)}), \pi^{(e)}) \notin R_{\mathrm{CSAT}}^f$.

Let v be the first such vertex (in the order that $\mathbb{E}$ extracts). Let i be the iteration in which $\mathbb{E}$ extracts v. Since $\mathcal{E}$ is **deterministic**, we know that $\widetilde{\mathcal{P}}$ also extracts v in the i-th iteration, and the corresponding index-instance-proof tuple $(\mathbb{i}_v, \mathbb{x}_v, \pi_v) := (\mathbb{i}, (\mathsf{ivk}, z^{(e)}), \pi^{(e)})$ for $e = (v, v')$ is the same as the one in $\mathbb{E}$. Hence, $(\mathbb{i}_v, \mathbb{x}_v, \mathbb{w}_v) \notin R_{\mathrm{CSAT}}$ by the argument above. Moreover, since v is the first such vertex, we can deduce that $\mathcal{V}^f(\mathbb{i}, (\mathsf{ivk}, z^{(e)}), \pi^{(e)}) = 1$. If $(\mathbb{i}_v, \mathbb{x}_v, \pi_v)$ is the tuple output by $\widetilde{\mathcal{P}}$, then from Definition 7,

$$\Pr\left[\begin{array}{c}\phi \text{ is } (\Phi, N, D, M, S, Q)\text{-compatible} \\ \wedge\ |z_{\mathrm{out}}| \leq l \\ \wedge\ \mathbb{V}^f(\mathbb{i}_{\mathbb{V}\mathbb{k}}, z_{\mathrm{out}}, \mathbb{\Pi}_{\mathrm{out}}) = 1 \\ \wedge\ (\mathrm{T} \text{ is not } (\phi, f)\text{-compliant} \vee \mathsf{out}(\mathrm{T}) \neq z_{\mathrm{out}})\end{array}\right] \leq \Pr\left[\begin{array}{c}|\mathbb{i}| \leq n \\ \wedge\ |\mathbb{x}| \leq k \\ \wedge\ (\mathbb{i}, \mathbb{x}, \mathbb{w}) \notin R \\ \wedge\ \mathcal{V}^f(\mathsf{ivk}, \mathbb{x}, \pi) = 1\end{array}\right]$$
$$\leq \kappa_{\mathsf{ARG}}(\lambda, \mathsf{q}_{\widetilde{\mathcal{P}}}, \mathsf{s}_{\widetilde{\mathcal{P}}}, n, k)\ ,$$

where $n := \mathsf{nsize}(\lambda, S, M, l)$ ($\mathsf{nsize}(\cdot)$ is the circuit size defined in Lemma 1) and $k := |\mathsf{ivk}| + l$.

We are left to show that $\widetilde{\mathcal{P}}$ outputs $(\mathbb{i}_v, \mathbb{x}_v, \pi_v)$. We proceed in two steps.

– Fix any $j < i$. We argue that $\widetilde{\mathcal{P}}$ does not output at iteration j. Let v_j be the vertex extracted at iteration j, and let $(\mathtt{i}_j, \mathtt{x}_j, \pi_j, \mathtt{w}_j)$ be the corresponding index, instance, proof, and witness of v_j. Let e_j be the unique outgoing edge of v_j, and let $v_{j,1}, \ldots, v_{j,M}$ be the child-vertices of v_j. Since we assume that v is the first vertex that "breaks" the strong compliance of T, it must be the case that
  - $\phi^f(z^{(e_j)}, w_{\mathrm{loc}}^{(e_j)}, (z^{(v_{j,1},v_j)}, \ldots, z^{(v_{j,M},v_j)})) = 1$; and
  - $\mathcal{V}^f(\mathsf{ivk}, (\mathsf{ivk}, z^{(v_{j,k},v_j)}), \pi^{(v_{j,k},v_j)}) = 1$ for all $k \in [M]$.

  Thus, for all $k \in [M]$ we know that $\mathsf{VerifyProof}(\mathsf{ivk}, (\mathsf{ivk}, z^{(v_{j,k},v_j)}), \pi^{(v_{j,k},v_j)}) = 1$, which follows since $\mathcal{V}^f(\mathsf{ivk}, (\mathsf{ivk}, z^{(v_{j,k},v_j)}), \pi^{(v_{j,k},v_j)}) = 1$. Therefore, the checks in Item 11i cannot pass, and $\widetilde{\mathcal{P}}$ does not output at iteration j.
– We argue that $\widetilde{\mathcal{P}}$ outputs $(\mathtt{i}_v, \mathtt{x}_v, \pi_v)$ at iteration i. Recall that v is the first vertex that "breaks" the strong compliance of T. Therefore, according to Definition 18, we distinguish between following two cases:
  - $\phi^f(z^{(e)}, w_{\mathrm{loc}}^{(v)}, (z^{(v_1,v)}, \ldots, z^{(v_M,v)})) \neq 1$: In this case, the first check in Item 11i passes, and $\widetilde{\mathcal{P}}$ outputs $(\mathtt{i}_v, \mathtt{x}_v, \pi_v)$ as desired.
  - $\mathcal{V}^f(\mathsf{ivk}, (\mathsf{ivk}, z^{(v_k,v)}), \pi^{(v_k,v)}) \neq 1$ for some $k \in [M]$: We have either $\mathsf{VerifyProof}(\mathsf{ivk}, (\mathsf{ivk}, z^{(v_i,v)}), \pi^{(v_i,v)}) \neq 1$, which makes the second check in Item 11i pass as desired; or $\mathsf{VerifyTrace}^f(\pi^{(v_k,v)}) \neq 1$, which cannot happen if the PCD verifier $\mathbb{V}$ accepts $(\mathtt{ivk}, z_{\mathrm{out}}, \mathbb{\Pi}_{\mathrm{out}})$.

5.2 Extraction Time Bound

We prove the upper bound on extraction time claimed in Theorem 5.

Proof. For every depth d and i-th vertex at depth d, we construct an argument prover $\widetilde{\mathcal{P}}_{d,i}$ for the invocation of the argument extractor $\mathcal{E}$ for the i-th vertex at depth d. (We consider v_1 to be the 0-th vertex at depth 1.)

Construction 7. *The argument prover $\widetilde{\mathcal{P}}_{1,0}$ corresponds to the first invocation of $\mathcal{E}$ in $\mathbb{E}$.*

$\widetilde{\mathcal{P}}^f_{1,0}(\mathsf{pp}, \mathsf{ai})$*:*
1. *Set* $\mathbb{pp} := \mathsf{pp}$.
2. *Run* $(\phi, z_{\mathrm{out}}, \mathbb{\Pi}_{\mathrm{out}}) \leftarrow \widetilde{\mathbb{P}}^f(\mathbb{pp}, \mathsf{ai})$.
3. *Run* $(\mathtt{i}\mathbb{pk}, \mathtt{i}\mathbb{vk}) \leftarrow \mathbb{I}^f(\mathbb{pp}, \phi)$.
4. *Parse* $\mathtt{i}\mathbb{vk}$ *as* ivk.
5. *Parse* $\mathbb{\Pi}_{\mathrm{out}}$ *as* π_{out}.
6. *Set* $\mathtt{i} := C_{\mathcal{V},\phi}^{(\lambda,M,n,k)}$.
7. *Output* $(\mathtt{i}, (\mathsf{ivk}, z_{\mathrm{out}}), \pi_{\mathrm{out}})$.

Construction 8. *The (recursively defined) argument prover $\widetilde{\mathcal{P}}_{d,i}$ corresponds to the invocation of $\mathcal{E}$ for the i-th vertex at depth d for $d > 1$ and $0 \leq i < M^{d-1}$ in $\mathbb{E}$.*

$\widetilde{\mathcal{P}}^f_{d,i}(\mathsf{pp}, \mathsf{ai})$*:*
1. *Let* $(\mathrm{parent}, \mathrm{pos}) := (\lfloor i/M \rfloor, i \bmod M)$.
2. *Run* $(\mathtt{i}, \mathtt{x}, \pi) \xleftarrow{\mathsf{tr}} \widetilde{\mathcal{P}}^f_{d-1,\mathrm{parent}}(\mathsf{pp}, \mathsf{ai})$.

3. *If* $(\mathbb{i}, \mathbb{x}, \pi) = \bot$*, halt and output* $\bot$.
4. *Run the argument extractor* $\mathbb{w} \leftarrow \mathcal{E}(\mathsf{pp}, \mathbb{i}, \mathbb{x}, \pi, \mathsf{tr})$.
5. *Parse* $\mathbb{w}$ *as* $(w_{\mathrm{loc}}, \boldsymbol{z}_{\mathrm{in}}, \boldsymbol{\Pi}_{\mathrm{in}})$.
6. *Parse* $\mathbb{x}$ *as* (ivk, z).
7. *If* $\boldsymbol{z}_{\mathrm{in}}[\mathrm{pos}] = \bot$*, output* $\bot$.
8. *Otherwise, output* $(\mathbb{i}, (\mathsf{ivk}, \boldsymbol{z}_{\mathrm{in}}[\mathrm{pos}]), \boldsymbol{\Pi}_{\mathrm{in}}[\mathrm{pos}])$.

The running time of $\mathbb{E}$ can be upper bounded in terms of the running time of $\mathcal{E}$ and extra processing time for the outputs of $\mathcal{E}$. Specifically, for every d and i, let $\mathsf{q}_{d,i}$ be the number of queries made by $\widetilde{\mathcal{P}}_{d,i}$, $n_{d,i}$ and $k_{d,i}$ be the sizes of the index and instance output by $\widetilde{\mathcal{P}}_{d,i}$. The running time of the argument extractor $\mathcal{E}$ when invoked for the i-th vertex v at depth d is, by definition, at most

$$\mathsf{t}_{\mathsf{ARG}}(\lambda, \mathsf{q}_{d,i}, n_{d,i}, k_{d,i}) \ .$$

We now upper bound $\mathsf{q}_{d,i}, n_{d,i}, k_{d,i}$.

Index Size and Instance Size. Every $\widetilde{\mathcal{P}}_{d,i}$ outputs an instance of the form $\mathbb{x} = (\mathsf{ivk}, z)$, so $k_{d,i} \leq k = |\mathsf{ivk}(\lambda, n)| + l$. Every $\widetilde{\mathcal{P}}_{d,i}$ outputs the index $\mathbb{i} = C^{(\lambda,M,n,k)}_{\mathcal{V},\phi}$ (the recursive circuit), whose size is at most

$$\mathsf{csize}\,(\lambda, S, M, l, n) \coloneqq S + O(M \cdot l) + M \cdot \mathsf{vsize}(\lambda, n, |\mathsf{ivk}(\lambda, n)| + l) \ .$$

Above:

- S is an upper bound on the predicate circuit size $|\phi|$;
- $O(M \cdot l)$ bounds the cost of going over all incoming messages;
- $\mathsf{vsize}(\lambda, n, k)$ is the size of $\mathcal{V}^{(\lambda,n,k)}$.

The below lemma gives a bound on the above expression, showing that $n_{d,i} \leq n = \mathsf{nsize}\,(\lambda, S, M, l)$.

Lemma 1 ([20, Lemma 11.8]). *Suppose that for every security parameter* $\lambda \in \mathbb{N}$ *and message size bound* $l \in \mathbb{N}$*, the ratio of verifier circuit size to index size* $\frac{\mathsf{vsize}(\lambda,n,|\mathsf{ivk}(\lambda,n)|+l)}{n}$ *is monotonically decreasing in* n*. Then there exists a size function* $\mathsf{nsize}\,(\lambda, S, M, l)$ *such that*

$$\forall \lambda, S, M, l \in \mathbb{N}, \mathsf{csize}\,(\lambda, S, M, l, \mathsf{nsize}\,(\lambda, S, M, l)) \leq \mathsf{nsize}\,(\lambda, S, M, l) \ .$$

Query Bound on the Argument Adversary. The malicious PCD prover $\widetilde{\mathbb{P}}$ makes $\mathsf{q}_{\mathbb{P}}$ queries to the oracle, and the PCD indexer $\mathbb{I}$ makes $\mathsf{q}_{\mathbb{I}}$ queries to the oracle. Therefore, the argument prover $\widetilde{\mathcal{P}}_{1,0}$ makes $\mathsf{q}_{\mathbb{P}} + \mathsf{q}_{\mathbb{I}}$ queries to the oracle; in fact, every argument prover $\widetilde{\mathcal{P}}_{d,i}$ makes $\mathsf{q}_{\mathbb{P}} + \mathsf{q}_{\mathbb{I}}$ queries to the oracle.

Therefore, the cost of running the argument extractor $\mathcal{E}$ across all its invocations in $\mathbb{E}$ is at most

$$N \cdot \mathsf{t}_{\mathsf{ARG}}(\lambda, \mathsf{q}_{d,i}, n_{d,i}, k_{d,i}) \ .$$

Additionally, during the execution of $\mathbb{E}$, for every vertex in the transcript, the extractor $\mathbb{E}$ reads the input messages and attaches the augmented label to the vertex, which takes at most $\mathsf{poly}(l, \mathsf{arglen}(n_{d,i}, k_{d,i}))$ bit operations, where $\mathsf{arglen}(n_{d,i}, k_{d,i})$ is the length of the argument proof in the label of the vertex. Also, the basic pop and push operations for a queue take at most $O(\log N)$ steps since there are at most $\mathsf{tsize}(\phi) \leq N$ extracted vertices. Therefore, the overhead is at most $N \cdot \mathsf{poly}(\log N, \log M, l, \mathsf{arglen}(n_{d,i}, k_{d,i}))$.

In conclusion, $\mathbb{E}$ runs in time

$$N \cdot (\mathsf{poly}(\log N, \log M, l, \mathsf{arglen}(n, k)) + \mathsf{t}_{\mathsf{ARG}}(\lambda, \mathsf{q}_{\tilde{\mathbb{P}}} + \mathsf{q}_{\mathbb{I}}, n, k)) \quad . \tag{2}$$

Acknowledgments. We thank Sarah Bordage for valuable discussions in early stages of this work. We thank Giacomo Fenzi, Christian Knabenhans, and Giorgio Seguini for valuable feedback and comments on earlier drafts of this paper. Ziyi Guan is partially supported by the Ethereum Foundation. Eylon Yogev is supported by the Israel Science Foundation (Grant No. 2302/22), European Research Union (ERC, CRYPTOPROOF, 101164375), and by an Alon Young Faculty Fellowship. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Barbara, A., Chiesa, A., Guan, Z.: Relativized succinct arguments in the ROM do not exist. Cryptology ePrint Archive, Paper 2024/728 (2024). https://eprint.iacr.org/2024/728
2. Bartusek, J., Bronfman, L., Holmgren, J., Ma, F., Rothblum, R.D.: On the (in)security of Kilian-based SNARGs. In: Proceedings of the 17th Theory of Cryptography Conference, pp. 522–551. TCC 2019 (2019)
3. Beal, J., Fisch, B.: Derecho: privacy pools with proof-carrying disclosures. Cryptology ePrint Archive, Paper 2023/273 (2023). https://eprint.iacr.org/2023/273
4. Bellare, M., Rogaway, P.: Random oracles are practical: A paradigm for designing efficient protocols. In: Proceedings of the 1st ACM Conference on Computer and Communications Security, pp. 62–73. CCS 1993 (1993)
5. Ben-Sasson, E., Bentov, I., Horesh, Y., Riabzev, M.: Scalable zero knowledge with no trusted setup. In: Proceedings of the 39th Annual International Cryptology Conference, pp. 733–764. CRYPTO 2019 (2019)
6. Ben-Sasson, E., Chiesa, A., Spooner, N.: Interactive oracle proofs. In: Proceedings of the 14th Theory of Cryptography Conference, pp. 31–60. TCC 2016-B (2016)
7. Ben-Sasson, E., Chiesa, A., Tromer, E., Virza, M.: Scalable zero knowledge via cycles of elliptic curves. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014. LNCS, vol. 8617, pp. 276–294. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-44381-1_16
8. Bitansky, N., Canetti, R., Chiesa, A., Tromer, E.: Recursive composition and bootstrapping for SNARKs and proof-carrying data. In: Proceedings of the 45th ACM Symposium on the Theory of Computing, pp. 111–120. STOC 2013 (2013)
9. Boneh, D., Drake, J., Fisch, B., Gabizon, A.: Halo infinite: proof-carrying data from additive polynomial commitments. In: Proceedings of the 41st Annual International Cryptology Conference, pp. 649–680. CRYPTO 2021 (2021)

10. Bonneau, J., Meckler, I., Rao, V., Shapiro, E.: Coda: Decentralized cryptocurrency at scale. IACR Cryptology ePrint Archive, Report 2020/352 (2020)
11. Bowe, S., Grigg, J., Hopwood, D.: Halo: Recursive proof composition without a trusted setup. Cryptology ePrint Archive, Report 2019/1021 (2019)
12. Bünz, B., Chen, B.: ProtoStar: generic efficient accumulation/folding for special-soundprotocols. In: Proceedings of the 29th International Conference on the Theory and Application of Cryptology and Information Security, pp. 77–110. ASIACRYPT 2023 (2023)
13. Bünz, B., Chiesa, A., Lin, W., Mishra, P., Spooner, N.: Proof-carrying data without succinct arguments. In: Proceedings of the 41st Annual International Cryptology Conference, pp. 681–710. CRYPTO 2021 (2021)
14. Bünz, B., Chiesa, A., Mishra, P., Spooner, N.: Proof-carrying data from accumulation schemes. In: Proceedings of the 18th Theory of Cryptography Conference, pp. 1–18. TCC 2020 (2020)
15. Canetti, R., Goldreich, O., Halevi, S.: The random oracle methodology, revisited. J. ACM **51**(4), 557–594 (2004)
16. Chen, M., Chiesa, A., Gur, T., O'Connor, J., Spooner, N.: Proof-carrying data from arithmetized random oracles. In: Proceedings of the 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, pp. 379–404. EUROCRYPT 2023 (2023)
17. Chen, M., Chiesa, A., Spooner, N.: On succinct non-interactive arguments in relativized worlds. In: Proceedings of the 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques. EUROCRYPT 2022 (2022)
18. Chen, W., Chiesa, A., Dauterman, E., Ward, N.P.: Reducing participation costs via incremental verification for ledger systems. Cryptology ePrint Archive, Report 2020/1522 (2020)
19. Chiesa, A., Liu, S.: On the impossibility of probabilistic proofs in relativized worlds. In: Proceedings of the 11th Innovations in Theoretical Computer Science Conference. ITCS 2020 (2020)
20. Chiesa, A., Ojha, D., Spooner, N.: Fractal: Post-quantum and transparent recursive proofs from holography. In: Proceedings of the 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, pp. 769–793. EUROCRYPT 2020 (2020)
21. Chiesa, A., Tromer, E.: Proof-carrying data and hearsay arguments from signature cards. In: Proceedings of the 1st Symposium on Innovations in Computer Science, pp. 310–331. ICS 2010 (2010)
22. Chiesa, A., Tromer, E., Virza, M.: Cluster computing in zero knowledge. In: Proceedings of the 34th Annual International Conference on Theory and Application of Cryptographic Techniques, pp. 371–403. EUROCRYPT 2015 (2015)
23. Chiesa, A., Yogev, E.: Subquadratic SNARGs in the random oracle model. In: Proceedings of the 41st Annual International Cryptology Conference, pp. 711–741. CRYPTO 2021 (2021)
24. Chiesa, A., Yogev, E.: Tight security bounds for Micali's SNARGs. In: Proceedings of the 19th Theory of Cryptography Conference, pp. 401–434. TCC 2021 (2021)
25. Chiesa, A., Yogev, E.: Building Cryptographic Proofs from Hash Functions (2024). https://github.com/hash-based-snargs-book
26. Chong, S., Tromer, E., Vaughan, J.A.: Enforcing language semantics using proof-carrying data. Cryptology ePrint Archive, Report 2013/513 (2013)
27. Ethereum. Zero-Knowledge Rollups (2023). https://ethereum.org/en/developers/docs/scaling/zk-rollups/
28. Fiore, D., Nitulescu, A.: On the (in)security of SNARKs in the presence of oracles. In: Proceedings of the 14th Theory of Cryptography Conference, pp. 108–138. TCC 2016-B (2016)
29. Gentry, C., Wichs, D.: Separating succinct non-interactive arguments from all falsifiable assumptions. In: Proceedings of the 43rd Annual ACM Symposium on Theory of Computing, pp. 99–108. STOC 2011 (2011)

30. Goldberg, L., Papini, S., Riabzev, M.: Cairo: a turing-complete STARK-friendly CPU architecture. IACR Cryptology ePrint Archive, Report 2021/1063 (2021)
31. Hall-Andersen, M., Nielsen, J.B.: On valiant's conjecture: impossibility of incrementally verifiable computation from random oracles. In: Proceedings of the 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques. EUROCRYPT 2023 (2023)
32. Kattis, A., Bonneau, J.: Proof of necessary work: Succinct state verification with fairness guarantees. In: Proceedings of the 27th Financial Cryptography and Data Security. FC 2023 (2023)
33. Kothapalli, A., Setty, S.: SuperNova: proving universal machine executions without universal circuits. Cryptology ePrint Archive, Paper 2022/1758 (2022)
34. Kothapalli, A., Setty, S., Tzialla, I.: Nova: recursive zero-knowledge arguments from folding schemes. In: Proceedings of the 42nd Annual International Cryptology Conference, pp. 359–388. CRYPTO 2022 (2022)
35. Matter Labs. zkSync v1.1 "Reddit Edition": Recursion (2020). https://blog.matter-labs.io/zksync-v1-1-reddit-edition-recursion-up-to-3-000-tps-subscriptionsand-more-fea668b5b0ff
36. Micali, S.: Computationally sound proofs. SIAM J. Comput. **30**(4), 1253–1298 (2000). preliminary version appeared in FOCS 1994
37. Naveh, A., Tromer, E.: PhotoProof: cryptographic image authentication for any set of permissible transformations. In: Proceedings of the 37th IEEE Symposium on Security and Privacy, pp. 255–271. S&P 2016 (2016)
38. O(1) Labs: Mina Cryptocurrency (2017). https://minaprotocol.com/
39. Paneth, O., Pass, R.: Incrementally verifiable computation via rate-1 batch arguments. In: Proceedings of the 63rd Annual IEEE Symposium on Foundations of Computer Science, pp. 1045–1056. FOCS 2022 (2022)
40. Polygon: The go fast machine: adding recursion to polygon zkEVM (2023). https://polygon.technology/blog/the-go-fast-machine-adding-recursion-to-polygon-zkevm
41. Polymer Labs: a tutorial on writing proofs with Plonky2 (2022). https://polymerlabs.medium.com/a-tutorial-on-writing-zk-proofs-with-plonky2-part-i-be5812f6b798
42. StarkWare Industries: Starkware: SHARP Verifier (2021). https://etherscan.io/address/0x47312450b3ac8b5b8e247a6bb6d523e7605bdb60
43. StarkWare Industries: Recursive STARKs (2022). https://medium.com/@starkware/recursive-starks-78f8dd401025
44. Tyagi, N., Fisch, B., Zitek, A., Bonneau, J., Tessaro, S.: VeRSA: Verifiable registries with efficient client audits from RSA authenticated dictionaries. In: Proceedings of the 29th ACM Conference on Computer and Communications Security. pp. 2793–2807. CCS '22 (2022)
45. Valiant, P.: Incrementally verifiable computation or proofs of knowledge imply time/space efficiency. In: Proceedings of the 5th Theory of Cryptography Conference, pp. 1–18. TCC 2008 (2008)

Author Index

E. Boyle and M. Mahmoody (Eds.): TCC 2024, LNCS 15365, p. 497, 2025.
https://doi.org/10.1007/978-3-031-78017-2