Studies in the Efficiency and (versus) Security of Cryptographic Tasks

Mohammad Mahmoody-Ghidary

A Dissertation
Presented to the Faculty
of Princeton University
in Candidacy for the Degree
of Doctor of Philosophy

Recommended for Acceptance
by the Department of
Computer Science
Advisers: Boaz Barak

September 2010

Abstract

In this thesis, we deal with the following questions: **(1)** How *efficient* a cryptographic algorithm can be while achieving a desired level of *security*? **(2)** Since mathematical conjectures like $\mathbf{P} \neq \mathbf{NP}$ are necessary for the possibility of secure cryptographic primitives in the standard models of computation: **(a)** Can we base cryptography solely based on the widely believed assumption of $\mathbf{P} \neq \mathbf{NP}$, or do we need stronger assumptions? **(b)** Which alternative nonstandard models offer us provable security *unconditionally*, while being implementable in real life?

First we study the question of security vs. efficiency in public-key cryptography and prove *tight* bounds on the efficiency of black-box constructions of key-agreement and (public-key) digital signatures that achieve a desired level of security using "random-like" functions. Namely, we prove that any key-agreement protocol in the random oracle model where the parties ask at most n oracle queries can be broken by an adversary who asks at most $O(n^2)$ oracle queries and finds the key with high probability. This improves upon the previous $\widetilde{O}(n^6)$-query attack of Impagliazzo and Rudich [98] and proves that a simple key-agreement protocol due to Merkle [118] is optimal. We also prove that any signature scheme in the random oracle model where the parties ask at most q oracle queries can be broken by an adversary who forges a signature by asking at most $2^{O(q)}$ oracle queries. This implies that a simple (one-time secure) signature scheme of Lamport [112] is optimal.

Next, we study the possibility of basing the security of cryptographic tasks on the (necessary) assumption that $\mathbf{NP} \neq \mathbf{BPP}$. We show that any (black-box) reduction for basing the security of a one-way function on (worst-case) hardness of **NP** implies that SAT is checkable. Whether SAT is checkable or not has been open for more than two decades since the notion of checkability was introduced by Blum and Kannan [23]. Then we study the possibility of basing the security of *specific*

cryptographic tasks/primitive on the hardness of **NP**. We show that doing so for the tasks of collision resistant hashing (or other primitives such as constant-round statistical commitment) implies that **co-NP** has a (single-prover) proof system with prover complexity $\mathbf{BPP}^{\mathbf{NP}}$ (which implies the checkability of SAT).

Finally, we study the possibility of achieving *statistical* security (without relying on computational assumptions) through the alternative model of interaction in which parties can exchange *tamper-proof* hardware tokens. We focus on the case where the tokens only encapsulate efficient circuits (and does not need to keep any state). The stateless property of the tokens gives advantage to the protocol both from a security perspective and also at the implementation level. We show that using stateless tokens one can *not* perform statistically secure oblivious transfer, but it is possible to achieve statistically hiding and binding commitment schemes and statistically secure zero-knowledge proof systems for **NP** with an efficient prover. Our protocols are secure even against malicious parties who might use *stateful* tokens during the execution of the protocol.

Acknowledgements

I would like to start by thanking my wonderful adviser Boaz Barak whose support and guidance has always been with me during my five years of PhD at Princeton. I learnt immensely from talking to and working with Boaz who not only did tell me where exactly to look/read to find what I was searching for, but also taught me how to attack a problem from all directions till finally taming it.

I thank the computer science department of Princeton and in particular the faculties and students of the theory group there who provided a friendly environment for my research and life in Princeton. I also thank the members of my thesis committee: Sanjeev Arora, Moses Charikar, Bernard Chazelle, and Russell Impagliazzo.

I was lucky to be at Princeton while David Xiao was also there and some of our enjoyable discussions led to some of the work here. I also thank Iftach Haitner for great discussions and our work together. I would like to thank Yuval Ishai, Amit Sahai and Vipul Goyal for the great time I spend during the spring of 2009 in their fascinating crypto group at UCLA which among others led to some of the results presented here. I thank Avi Wigderson for insightful discussions, specially at the early stages of my PhD studies. I thank Amin Shokrollahi for great discussions and a lovely summer in Lausanne. I also thank Thomas Holenstein, Benny Applebaum, and Mahdi Cheraghchi, which discussions with whom affected my work presented here directly or indirectly. I thank Salil Vadhan for very insightful comments and suggestions during the development of some of the work presented here.

Finally, I thankfully acknowledge Princeton's Wu prize and NSF grants CNS-0627526, CCF-0426582 and CCF-0832797 for supporting my research.

Contents

Chapter 1

Introduction and Preliminaries

1.1 Introduction

Due to the enormous growth in scale and complexity of electronic transaction systems, it is now even more crucial to design algorithms whose security against adversarial behavior is *provable* (at least based on clear and well-studied assumptions) in a mathematical way.

Ideally we would like to design cryptosystems which **(1)** are efficient and implementable in real life and **(2)** have a high level of security against adversarial behavior. Unfortunately, sometimes these two goals become competitive and achieving both might be challenging or even impossible.

Here, we first study the above question (about the security versus efficiency trade-off) for the tasks of key-agreement and digital signatures (which both fall into the category of public-key cryptography). Then we turn into studying the limits of our techniques to design cryptosystems based on the most trusted mathematical assumptions such as $\mathbf{P} \neq \mathbf{NP}$ or the existence of one-way functions. Being obstructed by barriers against our techniques in the standard model of interaction, an alternative is to

pursue designing secure protocols based on newer models of computation/interaction which have the potential of being obtained in real life (e.g., quantum computation or tamper-proof hardware). The last part of our thesis follows the last approach by studying the possibility of achieving unconditional (statistical) security in the "tamper-proof hardware" model. In this model parties are allowed to exchange hardware tokens where the receiver is only able to run the hardware in an input/output fashion—without understanding the content of the hardware. In the following I have described the results in this thesis briefly. For more details and background see the introduction sections of each chapter.

1.1.1 Security vs. Efficiency in Public-Key Cryptography

The constructions used in practice for public key digital signature and encryption are based on *structured* problems such as the integer factoring [51, 141, 144]. The non-trivial attacks such as the subexponential factoring algorithms [34] and of efficient quantum factoring algorithms [149] place a limit to how large the security of the scheme can be as a function of the key size. These attacks strongly motivate the study of whether we can achieve strongly secure public key cryptosystems based on *unstructured* "random-like" functions (e.g., the block-cipher AES or the hash function SHA-2).

Looking at our techniques in design of cryptographic protocols, most constructions and proofs in the cryptographic literature are black-box, so it is worthwhile to understand whether black-box reductions can base cryptographic primitives on **NP**-hardness, and this is the framework under which we study the possibility or impossibility results here. In Chapter 2 and Chapter 3 we give *tight* quantitative answers both to the case of key-agreement and digital signature schemes explained as follows.

Public Key Encryption / Key Agreement. In 1974 Merkle [118] suggested the first key agreement protocol which could be considered also as a public-key cryptosystem based on random-like functions. Unfortunately Merkle's scheme only achieved $\Theta(n^2)$ security where n is the number of times that the parties need to execute a "random-like" function. In their 1989 paper, Impagliazzo and Rudich showed that any construction of key agreement from a random function can achieve at best $n^6 \log n$ (black-box) security and it was left open whether Merkle's scheme is optimal or one can do better. In Chapter 2 we resolve this open question by showing that any such key agreement protocol can achieve at best $O(n^2)$ black-box security.

Digital Signatures. In Chapter 3 we show that any algorithm for digital signature for m-bit messages achieving black-box security $S = 2^q$ from a random function needs to evaluate the random function at least q times for sufficiently large $m \geq q$. The bound, even for a mild desired security like $S = 2^{1000}$ to sign 1000-bit messages, makes the system inefficient. This is in contrast to message authentication codes—the *private*-key analog of signatures —that can be realized with security $S = 2^q$ using only a *single* invocation of the random function over input/output length q. In fact, our result shows that the well known construction by Lamport [112] from 1979 for (one-time) digital signatures is optimal.

1.1.2 NP-Hard Cryptography

A holy grail in foundations of cryptography is to base cryptography on the minimal assumption of $\mathbf{P} \neq \mathbf{NP}$; namely, to show that $\mathbf{P} \neq \mathbf{NP}$ implies the existence of one-way functions, or, even more desirably, the existence of stronger cryptographic primitives such as collision-resistant hash functions or public-key cryptosystems. Other than the fact that $\mathbf{P} \neq \mathbf{NP}$ is necessary for the existence of one-way functions (and almost

all other cryptographic primitives [97, 131]), the former is a "worst-case" assumption while the latter is of "average-case" nature, hence making the first assumption much more desirable. In fact, this goal dates back to the seminal paper by Diffie and Hellman [47].

A black-box reduction (also known as, black-box proof of security) from the security of a cryptographic primitive to **NP**-hardness, is an efficient randomized oracle algorithm R such that given any oracle $\mathcal{O}$ that breaks the security of the cryptographic primitive, $R^{\mathcal{O}}$ solves SAT. The question of whether black-box reductions can be used to base cryptography on **NP**-hardness has been previously studied in [5, 27, 31, 54, 64, 135].

In Chapter 4 evidence that for some *specific* tasks (such as collision-resistant hashing), basing the security on **NP**-hardness is unlikely to be possible. We prove that achieving this goal is unlikely to be possible (or at least very hard to do) by showing that such **NP**-hard primitives would imply the existence of interactive proof systems for the class **co-NP** with $\mathbf{BPP}^{\mathbf{NP}}$ prover complexity. The known provers in the proof systems for the class **co-NP** require #**P** computation [115, 148].

In Chapter 5 we show that constructing **NP**-hard one-way functions (or even more generally basing the average-case hardness of **NP** on its worst-case hardness) implies that SAT, the language of satisfiable formulae, has a program checker. Whether SAT has a program checker or not has been open for more than two decades [23, 57]. This connection between checkability and **NP**-hard cryptography gives motivation to further study even specific checkers (e.g. of bounded adaptivity) for SAT. The existence of program checkers for SAT is equivalent to the existence of a two-prover proof system for **co-NP**, with honest provers in $\mathbf{BPP}^{\mathbf{NP}}$. Therefore the results of Chapter 4 and Chapter 5 are incomparable.

In Chapter 5 we also show that the well-known lattice problems such as GapSVP

(i.e., approximating the length of the shortest vector in a lattice of dimension up to the factor $n/\sqrt{n}$) which are used to build cryptography over the worst-case assumptions [3], are unlikely to be **NP**-hard even under a *randomized* reduction. It was previously known only for *deterministic* reductions [65].

1.1.3 Unconditional Security by Exchanging Stateless Hardware

It is known that achieving statistical security which does not rely on computational assumptions, in the standard model of computation/interaction is impossible for almost all cryptographic tasks [97, 131]. In that respect, what is the minimal amount of trust, or alternative models which allows us to achive unconditionally secure cryptography? Unconditional cryptography can be based on trusted two-party functionalities such as oblivious transfer [107, 139] or noisy channels [42], on bounded storage assumptions [116], on the presence of an honest majority [16, 41, 142], or even on the presence of a dishonest majority of *non-communicating* parties [15]. More recently, there has been a considerable amount of work on cryptographic protocols in which parties can generate and exchange tamper-proof hardware tokens. In this model it was shown that unconditionally secure commitments [123] or even general secure two-party computation [80] are possible, provided that the tokens can be *stateful*. In particular, stateful tokens can erase their secrets after being invoked. The subject of Chapter 6 is motivated by the goal of establishing unconditional feasibility results for cryptography using *stateless* hardware tokens. This question turns out to be related to the classical question of unconditional multi-prover zero-knowledge proofs, which we revisit in this work.

In the stateless token model (short for "stateless tamper-proof hardware token") the parties are allowed to construct efficient circuits based on their private randomness/input and the messages received from other parties. Then they can put this circuit inside a token (making it tamper-proof) and send it to another party. The receiver is allowed to ask any input from the token, but the model does not allow the party to get any more information from the received token. Ideally we would like to construct protocols which the honest behavior of the parties does not require any memory to be planted inside the tokens, while the protocol is secure against maliciously chosen stateful tokens.

In Chapter 6 we give an almost complete map of which basic cryptographic tasks are possible or impossible to achieve with unconditional statistical security, while the parties only exchange *stateless* hardware tokens. Our main building block in our positive results of this chapter is to construct an *Interactive Locking Scheme* (ILS) which is a protocol to simulate a secure vault (or more technically a "commitment scheme") by using a single stateless token and interaction. Then we use our ILS to get statistically secure zero-knowledge proof systems for the class **NP** which the prover only sends a single token to the verifier followed by some classical interaction. Our prover can be implemented efficiently given the witness for the language **NP**. On the other hand, we show that "general" statistically secure two-party computation—in particular the task of oblivious transfer (OT)—is not possible using stateless tokens. Perhaps surprisingly, we show that if one extends the model to allow parties building token *around* the tokens that they have received, then statistically secure OT, and therefore any two-party computation [107, 139] is possible.

1.2 Previous Publications

The results of this thesis previously have appeared in the following papers:

- Chapter 2: B. Barak and M. Mahmoody-Ghidary. *Merkle puzzles are optimal - an $O(n^2)$-query attack on any key exchange from a random oracle.* In International Cryptology Conference (Crpyto), 2009.

- Chapter 3: B. Barak and M. Mahmoody-Ghidary. *Lower-bounds on signatures from symmetric primitives.* In IEEE Symposium on Foundations of Computer Science (FOCS), 2007.

- Chapter 4: I. Haitner, M. Mahmoody, and D. Xiao. *A new sampling protocol and applications to basing cryptographic primitives on hardness of* **NP**.. In Computational Complexity Conference (CCC), 2010.

- Chapter 5: M. Mahmoody and D. Xiao. *On the power of randomized reductions and the checkability of* SAT. In Computational Complexity Conference (CCC), 2010.

- Chapter 6: V. Goyal, Y. Ishai, M. Mahmoody, and A. Sahai. *Interactive locking, zero-knowledge PCPs, and unconditional cryptography.* In International Cryptology Conference (Crypto), 2010.

1.3 Preliminaries

In this section we only mention the basic preliminaries which are needed across (or in more than two) chapters. The requirements that are specifically needed for the results of each chapter are described in the preliminary section of the same chapter.

1.3.1 Basics about Probability

By $\text{Supp}(X)$ we mean the support set of the random variable X. By $x \leftarrow X$ we mean that x is sampled according to the distribution of the random variable X. For event E defined over $\text{Supp}(X)$, by $(X \mid E)$ we denote the random variable X conditioned on the event E. For a set S, by U_S we mean the random variable with uniform distribution over S. For a set S, by $x \xleftarrow{\text{R}} S$ we mean $x \leftarrow U_S$. By U_n we mean the random variable uniformly distributed over $\{0,1\}^n$.

As is standard convention, when talking about an ensemble of random variables $\{X_n\}$ and event $E = \{E_n\}$ defined over $\{X_n\}$, we say E occurs with negligible probability if it occurs with probability $n^{-\omega(1)} = \text{neg}(n)$ and we say that it occurs with overwhelming probability if it occurs with probability $1 - \text{neg}(n)$.

Definition 1.3.1. For the random variables X and Y defined over the set U we let $\mathsf{SD}(X,Y)$ denote their *statistical distance* defined as $\mathsf{SD}(X,Y) = \max_{E \subseteq U} |\Pr[X \in E] - \Pr[Y \in E]|$. We also use $X \approx_\alpha Y$ and call X and Y α-close whenever $\mathsf{SD}(X,Y) \leq \alpha$. Two ensembles of distributions $\{X_n\}, \{Y_n\}$ over $\{0,1\}^{\text{poly}(n)}$ are *statistically indistinguishable* if $X_n \approx_{\text{neg}} Y_n$.

It is easy to see that the triangle inequality holds for the statistical distance: $\mathsf{SD}(X,Y) + \mathsf{SD}(Y,Z) \geq \mathsf{SD}(X,Z)$.

It also can be verified that the the statistical distance is the maximum advantage by which a statistical test (i.e., an algorithm A which outputs in $\{0,1\}$) can

distinguish between two random variables X and Y: $\mathsf{SD}(X,Y) = \max_A |\Pr[A(X) = 1] - \Pr[A(Y) = 1]|$.

Lemma 1.3.2. *Let $X = (X_1, X_2)$ and $Y = (Y_1, Y_2)$ be random variables defined over the set $U_1 \times U_2$ and suppose $\mathsf{SD}(X,Y) \leq \epsilon$. Let $Z = (Z_1, Z_2)$ be the random variable defined over $U_1 \times U_2$ as follows: Z_1 is sampled according to X_1 (defined by the distribution of X), and then Z_2 is sampled according to $(Y_2 \mid Y_1)$ conditioned on $Z_1 = Y_1$. Then it holds that $\mathsf{SD}(Y,Z) \leq \epsilon$ and therefore by the triangle inequality $\mathsf{SD}(X,Z) \leq 2\epsilon$.*

Proof. Let $\mathsf{SD}(Y,Z) \geq \epsilon$ and let A be the algorithm that distinguishes between Y and Z with advantage $\geq \epsilon$. Then one can distinguish between X and Y with advantage more than ϵ as well by the following algorithm. Given the input $W = (W_1, W_2)$ (which is either sampled according to X or Y) we take the first component W_1 and sample Y_2 conditioned on $Y_1 = W_1$ according to the distribution of Y. We then apply the test A over (W_1, Y_2). It is easy to see that the new test distinguishes between Y and X as well as A does between Y and Z. □

The following two lemmas are easy to verify.

Lemma 1.3.3. *Let X be a random variable distributed over the set $(S \times T) \cup \{\perp\}$, and suppose $\mathsf{SD}(X, U_{S\times T}) \leq \Pr[X = \perp] + \delta$. Let X_S be the random variable that equals $\perp$ if $X = \perp$ and equals to $s \in S$ if $X = (s,t)$. Then it holds that $\mathsf{SD}(X_S, U_S) \leq \Pr[X = \perp] + \delta$.*

Lemma 1.3.4. *Let X, Y be a random variables distributed over the set $S \cup \{\perp\}$ such that $\Pr[Y = \perp] = 0$ and $\mathsf{SD}(X,Y) \leq \Pr[X = \perp] + \delta$. Then for any event $T \subset S$ it holds that:*

$$\Pr[X \in T] = \Pr_{x \leftarrow X}[x \neq \perp \wedge x \in T] \leq \Pr[T] + \delta.$$

Lemma 1.3.5 (Lemma 6.4 of [98]). *Let $Z_1, \ldots, Z_i, \ldots$ be any sequence of random variables determined by a finite underlying random variable X, let E be any event for random variable X, and let $0 \leq p \leq 1$. Let B_j be the event that $\Pr_X[E(X) \mid Z_1, \ldots, Z_j] \geq p$, and let $B = \bigvee_j B_j$. Then it holds that $\Pr_X[E[X] \mid B] \geq p$.*

1.3.2 Interactive Algorithms

We assume that the reader is familiar with the notion of interactive Turing Machines introduced in [76] which here we call *interactive algorithms*. An interactive *protocol* is defined by two interactive algorithms A and B where each of A and B may have its own private input, private random tape, and output. By $\langle A, B \rangle (x)$ we denote the output of B when A and B interact on the common input x. We take the output 1 to denote an "accept" and the output 0 to denote a "reject". When x is clear from the context we might omit it from the notation (e.g., letting $\langle A, B \rangle$ denote $\langle A, B \rangle (x)$).

We use the standard notation A^π to denote an *oracle algorithm* A accessing the oracle π.

By an *efficient* algorithm we mean one which runs in polynomial time over its input length. We do not assume the interactive algorithms to be necessarily efficient unless explicitly mentioned. We always let n denote the length of the input x which will also serve as the security parameter when A and B are efficient.

A *round* consists of a message from A followed by another message from B assuming that A starts the protocol. The round complexity of the protocol (A, B) is the maximum number of rounds that A and B interact as a function of the input length n.

The *view* of an interactive algorithm consists of its input, its randomness and the answers received from the other interactive algorithm participating in the protocol.

1.3.3 AM Languages

A language L is in $\mathbf{AM}[k]$ if there exists a k-round interactive protocol for deciding L where the verifier V is efficient (i.e., polynomially bounded) and public coin (its messages are simply random coins). Namely, for every $x \in L \cap \{0,1\}^n$ it holds that $\Pr[V \text{ accepts in } \langle P, V\rangle(x)] \geq 1 - 2^{-n}$ and for every $x \in \{0,1\}^n \setminus L$ and any cheating (possibly inefficient) prover P^* it holds that $\Pr[V \text{ accepts in } \langle P^*, V\rangle(x)] \leq 2^{-n}$. If $k = 2$, then we simply write $L \in \mathbf{AM}$ (and call L an $\mathbf{AM}$ set or language). Finally, we abuse notation and say that (P, V) is an $\mathbf{AM}[k]$ protocol if it is a k-round, public-coin protocol with an efficient verifier.

We also consider the promise variant of an $\mathbf{AM}$ language.

Definition 1.3.6. Let $\mathsf{M} = (P, V)$ be an $\mathbf{AM}$ protocol. We say that M is a proof system for the *promise problem* (Y, N) (where $Y \cap N = \varnothing$) if the following holds.

- $Y = \bigcup_n Y_n$ where $Y_n = \{x \in \{0,1\}^n \colon \Pr[V \text{ accepts in } \langle P, V\rangle(x)] \geq 1 - 2^{-n}\}$.
- $N = \bigcup_n N_n$ where $N_n = \{x \in \{0,1\}^n \colon \forall P^*, \Pr[V \text{ rejects in } \langle P^*, V\rangle(x)] \geq 1 - 2^{-n}\}$.

We call Y_n the set of YES instances, N_n the set of NO instances and $T_n = \{0,1\}^n \setminus (Y_n \cup N_n)$ the set of *non-promise* instances of length n. We also let $T = \bigcup_n T_n$ to be the set of all non-promise inputs to M.

Note that a language $L \in \mathbf{AM}$ iff there exist a two-round public-coin protocol (with an efficient verifier) $\mathsf{M} = (P, V)$ with an empty non-promise set $T = \varnothing$.

Part I

Security vs. Efficiency in Public-Key Cryptography

Chapter 2

Merkel's Key-Agreement Protocol is Optimal

2.1 Introduction

In the 1970's Diffie, Hellman, and Merkle began to challenge the accepted wisdom that two parties cannot communicate confidentially over an open channel without first exchanging a secret key using some secure means. The first such protocol (at least in the open scientific community) was designed by Merkle in 1974 (although only published in 1978 [118]). Merkle's protocol allows two parties Alice and Bob to agree on a random number k that will not be known to an eavesdropping adversary Eve. It is described in Fig. 2.1.

[1]Merkle described his protocol using "puzzles" that can be implemented via some ideal cryptographic primitive; we describe the protocol in the case that the puzzles are implemented by a random oracle. We remark that in Merkle's original protocol Bob will try different random queries $y_1, y_2, \ldots$ without sending them to Alice until he finds y_j such that $f(y_j) \in \{a_1, \ldots, a_{10n}\}$ and send j — the index of the "puzzle" a_j — to Alice. The Protocol of Fig. 2.1 is a *symmetric* version of Merkle's protocol, and is similar to the protocol of [35] in the bounded storage model; see also discussion in [18].

Merkle's Key Exchange Protocol

Let n be the security parameter and $H : \{0,1\}^\ell \mapsto \{0,1\}^\ell$ where $\ell \gg \log n$ be a random function accessible for all parties. The protocol is as follows:

1. Alice chooses $10n$ random numbers $x_1, \ldots, x_{10n}$ in $[n^2]$ and sends $a_1, \ldots, a_{10n}$ to Bob where $a_i = H(x_i)$ (embed $[n^2]$ in $\{0,1\}^\ell$ in some canonical way).

2. Bob chooses $10n$ random numbers $y_1, \ldots, y_{10n}$ in $[n^2]$ and sends $b_1, \ldots, b_{10n}$ to Alice where $b_j = H(x_j)$.

3. With at least 0.9 probability, there will be at least one "collision" between Alice's and Bob's messages: a pair i, j such that $a_i = b_j$. Alice and Bob choose the lexicographically first such pair, and Alice sets $s_A = x_i$ as her secret, and Bob sets $s_B = y_j$ as his secret. If no collision occurred they will not choose any secret. Note that assuming $2^\ell \gg n^4$, H will be one to one on $[n^2]$ with very high probability and hence $H(x_i) = H(y_j)$ implies $x_i = y_j$.

Analysis: the collision is distributed uniformly in $[n^2]$ and thus an adversary making $o(n^2)$ queries to the oracle will find the secret with $o(1)$ probability.

Figure 2.1: Merkle's key exchange protocol. [1]

One problem with Merkle's protocol is that its security was only analyzed in the random oracle model which does not necessarily capture security when instantiated with a cryptographic one-way or hash function [38]. Recently, Biham, Goren and Ishai [18] took a step towards resolving this issue by providing a security analysis for Merkle's protocol under the concrete complexity assumption of existence of exponentially hard one-way functions. In particular, they proved that assuming there exist a one-way function that cannot be inverted with probability more than $2^{-\alpha n}$ by adversaries running in time $2^{\alpha n}$ for $\alpha \geq 1/2 - \delta$, there is a key exchange protocol in which Alice and Bob run in time n but any adversary whose running time is at most $n^{2-10\delta}$ has $o(1)$ chance of finding the secret. But the most serious issue with Merkle's protocol is that it only provides a *quadratic* gap between the running time of the honest parties and the adversary. Fortunately, not too long after Merkle's work, Diffie and Hellman [47] and later Rivest, Shamir, and Adleman [144] gave constructions for key exchange protocols that are conjectured to have *super-polynomial* (even subexponential) security. But because these and later protocols are based on certain algebraic computational problems, and so could perhaps be vulnerable to unforseen attacks using this algebraic structure, it remained an important question to show whether there exist key exchange protocols with superpolynomial security that use only a random oracle.[2] The seminal paper of Impagliazzo and Rudich [98] answered this question negatively by showing that every key exchange protocol using n queries in the random oracle model can be broken by an adversary asking $O(n^6 \log n)$ queries.[3] Since a random oracle is in particular a one-way function (with high proba-

[2]This is not to be confused with some more recent works such as [13], that combine the random oracle model with assumptions on the intractability of other problems such as factoring or the RSA problem to obtain more efficient cryptographic constructions.

[3]More accurately, [98] gave an $O(m^6 \log m)$-query attack where m is the maximum of the number of queries n and the number of communication rounds, though we believe their analysis could be improved to an $O(n^6 \log n)$-query attack. For the sake of simplicity, when discussing [98]'s results we will assume that $m = n$, though for our result we do not need this assumption.

bility), this implied that there is no construction of a key exchange protocol based on a one-way function with a proof of super-polynomial security that is of the standard black-box type (i.e., a proof that transforms an adversary breaking the protocol into an inversion algorithm for the one-way function that only uses the adversary and the function as black boxes). Indeed, that was the motivation behind their result.

Question and motivation. Impagliazzo and Rudich [98, Section 8] mention as an open question (which they attribute to Merkle) to find out whether their attack can be improved to $O(n^2)$ queries (hence showing the optimality of Merkle's protocol in the random oracle model) or there exist key exchange protocols in the random oracle model with $\omega(n^2)$ security. Beyond just being a natural question, it also has some practical and theoretical motivations. The practical motivation is that protocols with sufficiently large polynomial gap could be secure enough in practice — e.g., a key exchange protocol taking 10^9 operations to run and $(10^9)^6 = 10^{54}$ operations to break could be good enough for many applications.[4] In fact, as was argued by [18], as technology improves and honest users can afford to run more operations, such polynomial gaps only become more useful. Thus if known algebraic key exchange protocols were broken, one might look to polynomial-security protocol such as Merkle's for an alternative. Another motivation is theoretical— Merkle's protocol has very limited interaction (consisting of one round in which both parties simultaneously broadcast a message) and in particular it implies a public key encryption scheme. It is natural to ask whether more interaction can help achieve some polynomial advantage over this simple protocol. A third, less direct motivation comes from quantum computing. In one scenario in which some algebraic key exchange protocols will be broken— the

[4]Of course, these numbers are just an example and in practical applications the constant terms will make an important difference. We note though that the above constants are not ruled out by [98]'s attack, but are ruled out by our attack (taking number of operations to mean the number of calls to the oracle).

construction of practical quantum computers— Merkle's protocol will also fail to offer non-trivial security due to Grover's search algorithm [81]. Our results below suggest (though do not prove) that Merkle's protocol may be optimal in this setting also, and so there may not exist a fully classical key-exchange protocol based on a one-way function with a black-box proof of super-linear security for quantum adversaries. We note that using quantum communication there is an *information theoretically* secure key-exchange protocol [17], and moreover, very recently Brassard and Salvail [32] (independently observed by [18]) gave a quantum version of Merkle's protocol, showing that if Alice and Bob can use quantum computation (but classical communication), to obtain a key-exchange protocol with super-linear (i.e., $n^{3/2}$) security in the random oracle model against quantum adversaries.

2.1.1 Our Result

In this work we answer the above question of [98], by showing that every protocol in the random oracle model where Alice and Bob make n oracle queries can be broken with high probability by an adversary making $O(n^2)$ queries. That is, we prove the following:

Theorem 2.1.1. *Let* Π *be a two-party protocol in the random oracle model such that when executing* Π *the two parties Alice and Bob make at most* n *queries each, and their outputs are identical with probability at least* ρ*. Then for every* $0 < \delta < 1$*, there is an adversary Eve making* $(\frac{16n}{\delta})^2$ *queries to the oracle whose output agrees with Bob's output with probability at least* $\rho - \delta$*.*

To the best of our knowledge, no better bound than the $\widetilde{O}(n^6)$-query attack of [98] was previously known even in the case where one does not assume the one-way function is a random oracle (hence making the task of proving a negative result easier).

We note that similarly to previous black-box separation results, our adversary can be implemented efficiently in a relativized world where $\mathbf{P} = \mathbf{NP}$.

2.2 Our Techniques

The main technical challenge in proving such a result is the issue of *dependence* between the executions of the two parties Alice and Bob in a key exchange protocol. At first sight, it may seem that a computationally unbounded attacker that monitors all communication between Alice and Bob will trivially be able to find out their shared key. But the presence of the random oracle allows Alice and Bob to correlate their executions even without communicating (which is indeed the reason that Merkle's protocol achieves non-trivial security). Dealing with such correlations is the cause of the technical complexity in both our work and the previous work of Impagliazzo and Rudich [98]. We handle this issue in a different way than [98]. On a very vague high level our approach can be viewed as using more information about the structure of these correlations than [98] did. This allows us to analyze a more efficient attacking algorithm, that is more frugal with the number of queries it uses than the attacker of [98]. Below we provide a more detailed (though still high level) exposition of our technique and its relation to [98]'s technique.

2.2.1 Comparison with Previous Work of Impagliazzo and Rudich

We now review [98]'s attack and outline of analysis, and particularly the subtle issue of *dependence* between Alice and Bob that arises in both their work and ours. The main novelty of our work is the way we deal with this issue, which is different from the approach of [98]. We believe that this review of [98]'s analysis and the way it

compares to ours can serve as a useful introduction to our actual proof. However, no result of this section is used in the later sections, and so the reader should feel free at any time to skip ahead to Section 2.3 and 2.4 that contain our actual attack and its analysis.

Consider a protocol that consists of n rounds of interaction, where each party makes exactly one oracle query before sending its message. [98] called protocols of this type "normal-form protocols" and gave an $\widetilde{O}(n^3)$ attack against them (their final result was obtained by transforming every protocol into a normal-form protocol with a quadratic loss of efficiency). Even though without loss of generality the attacker Eve of a key exchange protocol can defer all of her computation till after the interaction between Alice and Bob is finished, it is conceptually simpler in both [98]'s case and ours to think of the attacker Eve as running concurrently with Alice and Bob. In particular, the attacker Eve of [98] performed the following operations after each round i of the protocol:

- If the round i is one in which Bob sent a message, then at this point Eve samples $1000n \log n$ random executions of Bob from the distribution $\mathcal{D}$ of Bob's executions that are consistent with the information that Eve has at that moment (communication transcript and previous oracle answers). That is, Eve samples a uniformly random tape for Bob and uniformly random query answers subject to being consistent with Eve's information. After each time that she samples an execution, Eve asks the oracle all the queries asked during this execution and records the answers. (Generally, the true answers will not be the same answers as the one Eve guessed when sampling the execution.)

- Similarly, if the round i is one in which Alice sent a message then Eve samples $1000n \log n$ executions of Alice and makes the corresponding queries.

Overall Eve will sample $\widetilde{O}(n^2)$ executions making a total of $\widetilde{O}(n^3)$ queries. It's not hard to see that as long as Eve learns all of the *intersection queries* (queries asked by both Alice and Bob during the execution) then she can recover the shared secret with high probability. Thus the bulk of [98]'s analysis was devoted to showing the following statement, denoted below by **(*)**: *With probability at least* 0.9 *Eve never fails, where we say that Eve* fails *at round i if the query made in this round by, say, Alice was asked previously by Bob but not by Eve.*

2.2.2 The Issue of Independence

At first look, it may seem that one could easily prove **(*)**. Indeed, **(*)** will follow by showing that at any round i, the probability that Eve fails in round i *for the first time* is at most $1/(10n)$. Now all the communication between Alice and Bob is observed by Eve, and if no failure has yet happened then Eve has also observed all the intersection queries so far. Because the answers for non-intersection queries are completely random and independent from one another it seems that Alice has no more information about Bob than Eve does, and hence if the probability that Alice's query q was asked before by Bob is more than $1/(10n)$ then this query q has probability at least $1/(10n)$ to appear in each one of Eve's sampled executions of Bob. Since Eve makes $1000n \log n$ such samples, the probability that Eve misses q would be bounded by $(1 - \frac{1}{10n})^{1000n \log n} \ll 1/(10n)$.

When trying to make this intuition into a proof, the assumption that Eve has as much information about Bob as Alice does translates to the following statement: conditioned on Eve's information, the distributions of Alice's view and Bob's view are *independent* from one another.[5] Indeed, if this statement was true then the above

[5]Readers familiar with the setting of communication complexity may note that this is analogous to the well known fact that conditioning on any transcript of a 2-party communication protocol results in a product distribution (i.e., combinatorial rectangle) over the inputs. However, things are

paragraph could be easily translated into a proof that [98]'s attacker is successful, and it wouldn't have been hard to optimize this attacker to achieve $O(n^2)$ queries. Alas, this statement is false. Intuitively the reason is the following: even the fact that Eve has not missed any intersection queries is some non-trivial information that Alice and Bob share and creates dependence between them.[6]

Impagliazzo and Rudich [98] dealt with this issue by a "charging argument" (see also Remark 2.2.1 below), where they showed that such dependence can be charged in a certain way to one of the executions sampled by Eve, in a way that at most n samples can be charged at each round (and the rest of Eve's samples are distributed correctly as if the independence assumption was true). This argument inherently required sampling at least n executions (each of n queries) per round, hence resulting in an $\Omega(n^3)$ attack.

2.2.3 Our Approach

We now describe our approach and how it differs from the previous proof of [98]. The discussion below is somewhat high level and vague, and glosses over some important details. Again, the reader is welcome to skip ahead at any time to Section 2.3 that contains the full description of our attack, and does not depend on this section in any way.

Our attacking algorithm follows the same general outline, but has two important differences from the attacker of [98]:

---

different in the presence of a random oracle.

[6] As a simple example for such dependence consider a protocol where in the first round Alice chooses x to be either the string 0^n or 1^n at random, queries the oracle H at x and sends $y = H(x)$ to Bob. Bob then makes the query 1^n and gets $y' = H(1^n)$. Now even if Alice chose $x = 0^n$ and hence Alice and Bob have no intersection queries, Bob can find out the value of x just by observing that $y' \neq y$. Still, an attacker must ask a non-intersection query such as 1^n to know if $x = 0^n$ or $x = 1^n$.

1. One *quantitative* difference is that while our attacker Eve also computes a distribution $\mathcal{D}$ of possible executions of Alice and Bob conditioned on her knowledge, she does *not* sample from $\mathcal{D}$ full executions and then ask the arising queries. Rather, she computes whether there is any *heavy query*— a string $q \in \{0,1\}^*$ that has probability more than, say, $1/(100n)$ of being queried in $\mathcal{D}$— and makes only such heavy queries.

   Intuitively, since Alice and Bob make at most $2n$ queries, the total expected number of heavy queries (and hence the query complexity of Eve) is bounded by $O(n^2)$. The actual analysis is more involved since the distribution $\mathcal{D}$ keeps changing as Eve learns more information through the messages she observes and query answers she receives. We omit the details in this high-level overview.

2. The *qualitative* difference between the two attackers is that we do not consider the same distribution $\mathcal{D}$ that was considered by [98]. Their attacker to some extent "pretended" that the conditional distributions of Alice and Bob are independent from one another. In contrast, we define our distribution $\mathcal{D}$ to be the *real* distribution of Alice and Bob, where there could be dependencies between them. Thus to sample from our distribution $\mathcal{D}$ one would need to sample a *pair* of executions of Alice and Bob (random tapes and oracle answers) that are *jointly consistent* with one another and Eve's current knowledge. Another (less important) point is that the distribution $\mathcal{D}$ computed by Eve at each point in time will be conditioned not only on Eve's knowledge so far, but also on the event that she has not failed until this point.

The main challenge in the analysis is to prove that the attack is *successful*, that is that the statement **(*)** above holds, and in particular that the probability of failure at each round (or more generally, at each query of Alice or Bob) is bounded by, say,

$1/(10n)$. Once again, things would have been easy if we knew that the distribution $\mathcal{D}$ of the possible executions of Alice and Bob conditioned on Eve's knowledge (and not having failed so far) is a *product distribution*, and hence Alice has no more information on Bob than Eve has. While this is not generally true, we show that in our attack this distribution is *close to being a product distribution*, in a precise sense we define below.

At any point in the execution, fix Eve's current information about the system and define a bipartite graph G whose left-side vertices correspond to possible executions of Alice that are consistent with Eve's information and right-side vertices correspond to possible executions of Bob consistent with Eve's information. We put an edge between two executions A and B if they are consistent with one another and moreover if they do not represent an execution in which Eve *failed* prior to this point (i.e., there is no intersection query that is asked in both executions A and B but not by Eve). The distribution $\mathcal{D}$ that our attacker Eve considers can be thought of as choosing a random edge in the graph G. (Note that the graph G and the distribution $\mathcal{D}$ change at each point that Eve learns some new information about the system.) If G was the complete bipartite clique then $\mathcal{D}$ would be a product distribution. What we show is that G is *dense* in the sense that each vertex is connected to most of the vertices on the other side. We show that this implies that Alice's probability of hitting a query that Bob asked before is at most twice the probability that Eve does so if she chooses the most likely query based on her knowledge.

The bound on the degree is obtained by showing that G can be represented as a *disjointness graph*, where each vertex u is associated with a set $S(u)$ (from an arbitrarily large universe) and there is an edge between a left-side vertex u and a right-side vertex v if and only if $S(u) \cap S(v) = \varnothing$.[7] The definition of the graph G implies

[7]The set $S(u)$ will correspond to the queries that are made in the execution corresponding to u

that $|S(u)| \leq n$ for all vertices u. The definition of our attacking algorithm implies that the distribution obtained by picking a random edge $\{u, v\}$ and outputting $S(u) \cup S(v)$ is *light*, in the sense that there is no element q in the universe that has probability more than $1/(10n)$ of being contained in a set chosen from this distribution. We show that these properties together imply that each vertex is connected to most of the vertices on the other side, and so G is close to being a complete bipartite graph.

Remark 2.2.1 (Comparison with Previous Work)**.** One can also phrase the analysis of [98] in terms of a similar bipartite graph. Their argument involved fixing, say, Alice's execution which corresponds to fixing a left-side vertex u. As we noted above, if the degree of u is high (e.g., u is connected to most of the right side) then independence approximately holds and hence the probability that [98]'s attacker fails at this point is less than $1/(10n)$. The crucial component of [98]'s analysis was their observation that if the degree of u is low, then by taking a random vertex v on the right side and making all queries in the corresponding execution to v, one is likely to make progress in the sense that we learn a new query made in the execution corresponding to u. Now there are at most n new queries to learn, and hence if we sample much more than n queries then in most of them we're in the high degree case. This potential/charging argument inherently requires sampling *all* queries of the execution, rather than only the heavy ones, hence incurring a cost of at least n^2 queries *per round* or n^3 queries total.

In this work, we use the notion of "seminormal" protocols, which are the same as normal protocols with the difference that at each round Alice or Bob asks either either one or no query. It is easy to see that every protocol can be put in seminormal form without increasing the number of queries or loosing the security. We use this compilation only for the sake of analysis to show that our $O(n^2)$-attack works against

but *not* made by Eve.

the original form of the protocol.

In fact, using the idea of seminormal form, we observe that one can get an $\widetilde{O}(n^4)$ attack using [98]'s ideas saving a factor of n^2, but as we explained above, $\Omega(n^3)$ queries seems like an inherent limit to [98]'s approach.

2.3 Our Attacker

We consider a key exchange protocol Π in which Alice and Bob first toss coins r_A and r_B and then run Π using access to a random oracle H that is a random function from $\{0,1\}^\ell$ to $\{0,1\}^\ell$ for some $\ell \in \mathbb{N}$. We assume that the protocol proceeds in some finite number of rounds, and no party asks the same query twice. In round k, if k is odd then Alice makes some number of queries and sends a message to Bob (and then Eve asks some oracle queries), and if k is even then Bob makes some queries and sends a message to Alice (and then Eve asks some oracle queries). At the end of the protocol Alice obtains an output string s_A and Bob obtains an output string s_B. We assume that there is some constant $\rho > 0$ such that $\Pr[s_A = s_B] \geq \rho$, where the probability is over the coin tosses of Alice and Bob and the randomness of the oracle. We will establish Theorem 2.1.1 by proving that an attacker can make $O(n^2)$ queries to learn s_B with probability arbitrarily close to ρ.

In this section we describe an attacking algorithm that allows Eve to find a set of size $O(n^2)$ that contains all the queries asked by Alice and Bob in the random oracle model. This attack is analyzed in Section 2.4 to show that it is successful in finding all intersection queries and is efficient (i.e., will not ask more than $O(n^2)$ many queries). As was shown by Impagliazzo and Rudich, it not hard to use this set to obtain the actual secret.

2.3.1 Attacking Algorithm

We start by showing that an attacker can find all the *intersection queries* (those asked by both Alice and Bob) with high probability. It turns out that this is the main step in showing that an attacker can find the secret with high probability (see Section 2.5.2 below).

Theorem 2.3.1. *Let* Π *be a key exchange protocol in the random oracle model in which Alice and Bob ask at most* n *oracle queries each. Then for every* $0 < \delta < 1$ *there is an adversary Eve who has access to the messages sent between Alice and Bob and asks at most* $(\frac{13n}{\delta})^2$ *number of queries such that Eve's queries contain all the intersection queries of Alice and Bob with probability at least* $1 - \delta$.

Letting $\epsilon = \delta/13$, our attack can be described in one sentence as follows:

> *As long as there exists a string* q *such that conditioned on Eve's current knowledge and assuming that no intersection query was missed so far, the probability that* q *was asked in the past (by either Alice or Bob) is at least* ϵ/n*, Eve makes the query* q *to the oracle.*

To describe the attack more formally, we need to introduce some notation. We fix n to be the number of oracle queries asked by Alice and Bob and assume without loss of generality that all the queries are of length $\ell = \ell(n)$ for some $\ell \in \mathbb{N}$. We will make the simplifying assumption that the protocol is in *normal form*— that is, at every round of the protocol Alice or Bob make exactly one query to the oracle (and hence there are $2n$ rounds). Later in Section 2.5.1 we will show how our analysis extends to protocols that are not of this form. Below we often identify a distribution $\mathcal{D}$ with a random variable distributed according to $\mathcal{D}$.

Executions and the distribution $\mathcal{E}$. A (full) *execution* of Alice, Bob, and Eve can be described by a tuple (r_A, r_B, H) where r_A denotes Alice's random tape, r_B denotes Bob's random tape, and H is the random oracle (note that Eve is deterministic). We denote by $\mathcal{E}$ the distribution over (full) executions that is obtained by running the algorithms for Alice, Bob and Eve with uniformly chosen random tapes and a random oracle. A *partial execution* is an execution truncated at a certain point in time (that is, the transcripts contain only the oracle answers for queries that are asked up to that point). For any partial execution we denote by M the sequence of messages sent between Alice and Bob till that moment, and denote by $\mathcal{I}$ the set of oracle query/answer pairs known to Eve. We define *Alice's view* in the execution to be the tuple $A = (r_A, H_A, M)$ where r_A are Alice's coins and H_A is the concatenation of oracle answers to Alice's queries. Similarly Bob's view is the tuple $B = (r_B, H_B, M)$. Below we will only consider Alice's and Bob's view conditioned on a fixed value of M and hence we drop M from these tuples and let $A = (r_A, H_A)$ and $B = (r_B, H_B)$.

The distribution $\mathcal{E}(M, \mathcal{I})$. For $M = [m_1, \ldots, m_i]$ a sequence of i messages, and $\mathcal{I}$ a set of query/answer pairs, we denote by $\mathcal{E}(M, \mathcal{I})$ the distribution over the views (A, B) of Alice and Bob in partial executions up to the point in the system in which the i'th message is sent (by Alice or Bob), where the transcript of messages equals M and the set of query/answer pairs that Eve learns equals $\mathcal{I}$. For every $(M, \mathcal{I})$ that have nonzero probability to occur in the protocol, the distribution $\mathcal{E}(M, \mathcal{I})$ can be sampled by first sampling (r_A, r_B, H) at random conditioned on being consistent with $(M, \mathcal{I})$ and then deriving from this tuple Alice's and Bob's views: $A = (r_A, H_A)$ and $B = (r_B, H_B)$.[8]

[8]Note that we can verify that the pair $(M, \mathcal{I})$ has nonzero probability to occur in the protocol by simulating Eve's algorithm on the transcript M, checking that whenever Eve makes a query, this query is in $\mathcal{I}$, in which case we feed Eve with the corresponding answer (and verifying at the end that there are no "extra" queries in $\mathcal{I}$ not asked by Eve). However in our attack the pair $(M, \mathcal{I})$ will

The event $\mathsf{Good}(M,\mathcal{I})$ and the distribution $\mathcal{GE}(M,\mathcal{I})$. The event $\mathsf{Good}(M,\mathcal{I})$ is defined as the event over $\mathcal{E}(M,\mathcal{I})$ that all the intersection queries asked by Alice and Bob during the partial execution are in $\mathcal{I}$. More formally let $Q(A)$ (resp., $Q(B)$) be the set of queries asked by Alice (resp., Bob) which are specified by Alice's view A (resp., Bob's view B). Therefore $\mathsf{Good}(M,\mathcal{I})$ is the same as $Q(A) \cap Q(B) \subset Q(\mathcal{I})$ where $Q(\mathcal{I})$ is the set of queries of $\mathcal{I}$ (note that $\mathcal{I}$ is a set of query/answser *pairs*). We define the distribution $\mathcal{GE}(M,\mathcal{I})$ to be the distribution $\mathcal{E}(M,\mathcal{I})$ conditioned on $\mathsf{Good}(M,\mathcal{I})$.

Eve's algorithm. The attacker Eve's algorithm is specified as follows. It is parameterized by some constant $0 < \epsilon < 1/10$. At any point in the execution, if M is the sequence of messages Eve observed so far and $\mathcal{I}$ is the query/answer pairs she learned so far, Eve computes for every $q \in \{0,1\}^\ell$ the probability p_q that q appears as a query in a random execution in $\mathcal{GE}(M,\mathcal{I})$. If $p_q > \epsilon/n$ then Eve asks q from the oracle and adds q and its answer to $\mathcal{I}$. (If there is more than one such q then Eve asks the lexicographically first one.) Eve continues in this way until there is no additional query she can ask, at which point she waits until she gets new information (i.e., observes a new message sent between Alice and Bob).

Note that Eve's algorithm above may ask much more than n^2 queries. However, we will show that the probability that Eve asks more than n^2/ϵ^2 queries is bounded by $O(\epsilon)$, and hence we can stop Eve after asking this many queries without changing significantly her success probability.

always be generated by running the actual protocol and so we won't need to run such checks.

2.4 Analysis of Attack: Proof of Theorem 2.3.1

We now go over the proof of Theorem 2.3.1. For $i \in [2n]$, define the event Fail_i to be the event that the query made at the i'th round is an intersection query but is not contained in the set $\mathcal{I}$ of query/answer pairs known by Eve, and moreover that this is the first query satisfying this condition. Let the event $\mathsf{Fail} = \bigvee_i \mathsf{Fail}_i$ be the event that at some point an intersection query is missed by Eve, and let the event Long be that Eve makes more than n^2/ϵ^2 queries. By setting $\epsilon = \delta/13$ and stopping Eve after n^2/ϵ^2 queries, Theorem 2.3.1 immediately follows from the following two lemmas:

Lemma 2.4.1 (Attack is successful)**.** *For every i, $\Pr_{\mathcal{E}}[\mathsf{Fail}_i] \leq \frac{3\epsilon}{2n}$. Therefore by the union bound, $\Pr_{\mathcal{E}}[\mathsf{Fail}] \leq 3\epsilon$.*

Lemma 2.4.2 (Attack is efficient)**.** $\Pr_{\mathcal{E}}[\mathsf{Long}] \leq 10\epsilon$.

2.4.1 Success of Attack: Proof of Lemma 2.4.1

Lemma 2.4.1 follows from the following stronger result which is the main technical lemma of this chapter:

Lemma 2.4.3. *Let i be even and let $B = (r_B, H_B)$ be some fixing of Bob's view in an execution up to the i'th message sent by him, and let $M, \mathcal{I}$ be some fixing of the messages exchanged and query/answer pairs learned by Eve in this execution such that $\Pr_{\mathcal{E}(M,\mathcal{I})}[\mathsf{Good}(M,\mathcal{I}) \mid B] > 0$. Then it holds that*

$$\Pr_{\mathcal{GE}(M,\mathcal{I})}[\mathsf{Fail}_i \mid B] \leq \frac{3\epsilon}{2n}.$$

That is, the probability that Fail_i happens is at most $\frac{3\epsilon}{2n}$ conditioning on Eve's information equalling $(M,\mathcal{I})$, Bob's view of the execution equalling B and $\mathsf{Good}(M,\mathcal{I})$.

We first see why Lemma 2.4.3 implies Lemma 2.4.1.

Proof of Lemma 2.4.1 from Lemma 2.4.3. Lemma 2.4.3 implies that in particular for every even i,

$$\Pr_{\mathcal{E}}[\mathsf{Fail}_i \mid \mathsf{Good}_i] \leq \frac{3\epsilon}{2n},$$

where Good_i denotes the event $\mathsf{Good}(M, \mathcal{I})$ where $M, \mathcal{I}$ are Eve's information just before the i'th round. But since Fail_i is the event that Eve fails at round i *for the first time*, Fail_i implies Good_i and hence $\Pr_{\mathcal{E}}[\mathsf{Fail}_i] \leq \Pr_{\mathcal{E}}[\mathsf{Fail}_i \mid \mathsf{Good}_i]$, establishing the statement of Lemma 2.4.1 for every even i. By symmetry, the analog of Lemma 2.4.3 for odd i also holds with the roles of Alice and Bob reversed, completing the proof for all i. □

Proof of Lemma 2.4.3.

Product characterization. Lemma 2.4.3 would be easy if the distribution $\mathcal{GE}(M, \mathcal{I})$ would have been a *product distribution*, with the views of Alice and Bob independent from one another. Roughly speaking this is because in this case Bob has no more information than Eve on the queries Alice made in the past, and hence also from Bob's point of view, no query is more probable than ϵ/n to have been asked by Alice. Unfortunately this is not the case. However, we can show that the distribution $\mathcal{GE}(M, \mathcal{I})$ is equal to the distribution obtained by taking some product distribution $\mathcal{A} \times \mathcal{B}$ and conditioning it on the event $\mathsf{Good}(M, \mathcal{I})$. [9]

The intuition is that fixing $(M, \mathcal{I})$, the probability that certain computation for Alice and Bob happens is proportional to $2^{-l|H|}$ where H is the set of queries asked by (this specific) Alice or Bob, but not by Eve. In case Alice and Bob do not have intersection queries out of Eve's queries, the set H can be partitioned into Alice's private queries H_A and Bob's private queries H_B. Hence we have $2^{-l|H|} = 2^{-l|H_A|}2^{-l|H_B|}$. But

[9] A similar observation was made by [98], see Lemma 6.5 there.

note that $2^{-l|H_A|}$ (resp., $2^{-l|H_B|}$) is independent of Bob's (resp., Alice's) computation. More formally, we prove the following lemma:

Lemma 2.4.4 (Product characterization)**.** *For every $M, \mathcal{I}$ denoting Eve's information up to just before the i'th query, if $\Pr_{\mathcal{E}(M,\mathcal{I})}[\mathsf{Good}(M,\mathcal{I})] > 0$ there exist a distribution $\mathcal{A}$ (resp., $\mathcal{B}$) over Alice's (resp., Bob's) view up to that point such that the distribution $\mathcal{GE}(M,\mathcal{I})$ is the same as the product distribution $(\mathcal{A} \times \mathcal{B})$ conditioned on the event $\mathsf{Good}(M,\mathcal{I})$:*

$$\mathcal{GE}(M,\mathcal{I}) = (\mathcal{A} \times \mathcal{B}) \mid \mathsf{Good}(M,\mathcal{I}).$$

Proof. We will show that for every pair of Alice/Bob views (A, B) in the probability space $\mathcal{E}(M,\mathcal{I})$ that satisfy the event $\mathsf{Good}(M,\mathcal{I})$, $\Pr_{\mathcal{GE}(M,\mathcal{I})}[(A,B)] = c(M,\mathcal{I})\alpha_A\alpha_B$ where α_A depends only on A, α_B depends only on B and $c(M,\mathcal{I})$ depends only on $M,\mathcal{I}$. This means that if we let $\mathcal{A}$ be the distribution such that $\Pr_{\mathcal{A}}[A]$ is proportional to α_A, and $\mathcal{B}$ be the distribution such that $\Pr_{\mathcal{B}}[B]$ is proportional to α_B, then $\mathcal{GE}(M,\mathcal{I})$ is proportional (and hence equal to) the distribution $\mathcal{A} \times \mathcal{B} \mid \mathsf{Good}(M,\mathcal{I})$.

Because $(A,B) \in \mathrm{Supp}(\mathcal{GE}(M,I))$, if (A,B) happens, it makes the event $\mathsf{Good}(M,I)$ hold, and so we have

$$\Pr_{\mathcal{E}(M,\mathcal{I})}[(A,B)] = \Pr_{\mathcal{E}(M,\mathcal{I})}[(A,B) \wedge \mathsf{Good}(M,I)] = \Pr_{\mathcal{E}(M,\mathcal{I})}[\mathsf{Good}(M,\mathcal{I})] \Pr_{\mathcal{GE}(M,\mathcal{I})}[(A,B)].$$

On the other hand, by definition we have $\Pr_{\mathcal{E}(M,\mathcal{I})}[(A,B)] = \frac{\Pr_{\mathcal{E}}[(A,B,M,\mathcal{I})]}{\Pr_{\mathcal{E}}[(M,\mathcal{I})]}$, therefore it holds that

$$\Pr_{\mathcal{GE}(M,\mathcal{I})}[(A,B)] = \frac{\Pr_{\mathcal{E}}[(A,B,M,\mathcal{I})]}{\Pr_{\mathcal{E}}[(M,\mathcal{I})] \Pr_{\mathcal{E}(M,\mathcal{I})}[\mathsf{Good}(M,\mathcal{I})]}.$$

The denominator of the righthand side is only dependent on M and $\mathcal{I}$. The numerator is equal to

$$2^{-|r_A|}2^{-|r_B|}2^{-\ell|Q(A)\cup Q(B)\cup Q(\mathcal{I})|}.$$

The reason is that the necessary and sufficient condition that $(A=(r_A,H_A),B=(r_B,H_B),M,\mathcal{I})$ happens is that when we choose an execution (r'_A,r'_B,H') then $r'_A=r_A$, $r'_B=r_B$ and H is consistent on the queries in $Q(A)\cup Q(B)\cup Q(\mathcal{I})$ with the answers specified in $H_A,H_B,\mathcal{I}$. Note that this will ensure that Alice and Bob will indeed produce the transcript M. Let $\alpha_A=2^{-|r_A|}2^{-\ell|Q(A)\setminus Q(\mathcal{I})|}$ and $\beta_B=2^{-|r_B|}2^{-\ell|Q(B)\setminus Q(\mathcal{I})|}$. Since $(Q(A)\setminus Q(\mathcal{I}))\cap(Q(B)\setminus Q(\mathcal{I}))=\varnothing$, the numerator is equal to $2^{-|r_A|}2^{-|r_B|}2^{-\ell|Q(A)\cup Q(B)\cup Q(\mathcal{I})|}=\alpha_A\beta_B 2^{-\ell|Q(\mathcal{I})|}$. Thus indeed $\Pr_{\mathcal{GE}(M,\mathcal{I})}[(A,B)]=c(M,\mathcal{I})\alpha_A\beta_B$ where $c(M,\mathcal{I})$ only depends on $(M,\mathcal{I})$.

□

Graph characterization. This product characterization implies that we can think of $\mathcal{GE}$ as a distribution over random edges of some bipartite graph G. Using some insights on the way this graph is defined, and the definition of our attacking algorithm, we will show that every vertex in G is connected to most of the vertices on the other side. We then show that this implies that Bob's chance of asking a query outside of $\mathcal{I}$ that was asked before by Alice is bounded by $O(\epsilon/n)$.

More precisely, fixing $M,\mathcal{I}$ that contain Eve's view up to just before the i'th round, define a bipartite graph $G=(V_L,V_R,E)$ as follows. Every node $u\in V_L$ will have a corresponding view A_u of Alice that is in the support of the distribution $\mathcal{A}$ obtained from Lemma 2.4.4; we let the number of nodes corresponding to a view A be proportional to $\Pr_{\mathcal{A}}[A]$, meaning that $\mathcal{A}$ corresponds to the uniform distribution over the left-side vertices V_L. Similarly, every node $v\in V_R$ will have a corresponding view

of Bob B_v such that $\mathcal{B}$ corresponds to the uniform distribution over V_R. We define $Q_u = Q(A_u) \setminus Q(\mathcal{I})$ for $u \in V_L$ to be the set of queries *outside of* $\mathcal{I}$ that were asked by Alice in the view A_u, and define $Q_v = Q(B_u) \setminus Q(\mathcal{I})$ similarly. We put an edge in the graph between u and v (denoted by $u \sim v$) if and only if $Q_u \cap Q_v = \varnothing$. Lemma 2.4.4 implies that the distribution $\mathcal{GE}(M, \mathcal{I})$ is equal to the distribution obtained by letting (u, v) be a random edge of the graph G and choosing (A_u, B_v).

It turns out that this graph is *dense* (i.e., every vertex is connected to almost all other vertices in the other side). The proof has two steps. The first one is to show that such graphs are "highly connected" in the sense that removing any vertex v and its neighbors from the graph, remains a small fraction of the edges in the graph. The reason is that otherwise, there is a member of Q_v which is heavy and Eve should have asked that query. The second step is to show that this notion of connectivity would imply that the graph dense (whenever the graph is bipartite). More formally, we prove the following lemma:

Lemma 2.4.5. *Let $G = (V_L, V_R, E)$ be the graph above. Then for every $u \in V_L$, $d(u) \geq |V_R|(1 - 2\epsilon)$ and for every $v \in V_R$, $d(v) \geq |V_L|(1 - 2\epsilon)$ where $d(w)$ is the degree of the vertex w.*

Proof. We first show that for every $w \in V_L$, $\sum_{v \in V_R, w \not\sim v} d(v) \leq \epsilon |E|$. The reason is that the probability of vertex v being chosen when we choose a random edge is $\frac{d(v)}{|E|}$ and if $\sum_{v \in V_R, w \not\sim v} \frac{d(v)}{|E|} > \epsilon$, it means that $\Pr_{(u,v) \xleftarrow{R} E}[Q_w \cap Q_v \neq \varnothing] \geq \epsilon$. Hence because $|Q_w| \leq n$, by the pigeonhole principle there exists $q \in Q_w$ such that $\Pr_{(u,v) \xleftarrow{R} E}[q \in Q_v] \geq \epsilon/n$. But this is a contradiction, because then q should be in $\mathcal{I}$ by the definition of the attack and hence cannot be in Q_w. The same argument shows that for every $w \in V_R$, $\sum_{u \in V_L, u \not\sim w} d(u) \leq \epsilon |E|$. Thus for every vertex $w \in V_L \cup V_R$, $\left|E^{\not\sim}(w)\right| \leq \epsilon |E|$ where $E^{\not\sim}(w)$ denotes the set of edges that are not adjacent to any neighbor of w (i.e., $E^{\not\sim}(w) = \{(u, v) \in E \mid u \not\sim w \wedge w \not\sim v\}$). Now the following claim proves the lemma.

Claim 2.4.6. *Let $G = (V_L, V_R, E)$ be a nonempty bipartite graph such that for every vertex w, $\left|E^{\not\sim}(w)\right| \leq \epsilon\,|E|$ for $\epsilon \leq 1/2$, then for all $u \in V_L, d(u) \geq |V_R|(1-2\epsilon)$ and for every $v \in V_R$, $d(v) \geq |V_L|(1-2\epsilon)$.*

Proof. Let $d_L = \min\{d(u) \mid u \in V_L\}$ and $d_R = \min\{d(v) \mid v \in V_R\}$. By switching the left and right sides if necessary, we may assume without loss of generality that **(*):** $\frac{d_L}{|V_R|} \leq \frac{d_R}{|V_L|}$. Thus it suffices to prove that $1-2\epsilon \leq \frac{d_L}{|V_R|}$. Suppose $1-2\epsilon > \frac{d_L}{|V_R|}$, and let $u \in V_L$ be the vertex that $d(u) = d_L < (1-2\epsilon)\,|V_R|$. Because for all $v \in V_R$ we have $d(v) \leq |V_L|$, thus using **(*)** we see that $|E^{\sim}(u)| \leq d_L\,|V_L| \leq d_R\,|V_R|$ where $E^{\sim}(u) = E \setminus E^{\not\sim}(u)$. On the other hand since we assumed that $d(u) < (1-2\epsilon)|V_R|$, there are more than $2\epsilon|V_R|d_R$ edges in $E^{\not\sim}(u)$, meaning that $|E^{\sim}(u)| < \left|E^{\not\sim}(u)\right|/(2\epsilon)$. But this implies

$$\left|E^{\not\sim}(u)\right| \leq \epsilon|E| = \epsilon\left(\left|E^{\not\sim}(u)\right| + |E^{\sim}(u)|\right) < \epsilon\left|E^{\not\sim}(u)\right| + \left|E^{\not\sim}(u)\right|/2,$$

which is a contradiction for $\epsilon < 1/2$ because the graph G is nonempty. □

□

Proof of Lemma 2.4.3 from Lemmas 2.4.4 and 2.4.5. Let $B, M, \mathcal{I}$ be as in Lemma 2.4.3 and q be Bob's query which is fixed now. By Lemma 2.4.4, the distribution $\mathcal{GE}(M, I)$ conditioned on getting B as Bob's view is the same as $(\mathcal{A} \times \mathcal{B})$ conditioned on $\mathsf{Good}(M, \mathcal{I}) \wedge (\mathcal{B} = B)$. By the definition of the bipartite graph $G = (V_L, V_R, E)$ it is the same as choosing a random edge $(u, v) \xleftarrow{R} E$ conditioned on $B_v = B$ and choosing (A_u, B_v). We prove Lemma 2.4.3 even conditioned on fixing v such that $B_v = B$. Now the distribution on Alice's view is the same as choosing $u \xleftarrow{R} N(v)$ to be a random neighbor of v and choosing A_u. Let $S = \{u \in V_L \mid q \in A_u\}$.

Then it holds that

$$\Pr_{u \xleftarrow{R} N(v)}[q \in A_u] \leq \frac{|S|}{d(v)} \leq \frac{|S|}{(1-2\epsilon)|V_L|} \leq \frac{|S|\,|V_R|}{(1-2\epsilon)\,|E|} \leq \frac{\sum_{u \in S} d(u)}{(1-2\epsilon)^2\,|E|} \leq \frac{\epsilon}{(1-2\epsilon)^2 n} < \frac{3\epsilon}{2n}.$$

The second and fourth inequalities are because of Lemma 2.4.5. The third one is because $|E| \leq |V_L|\,|V_R|$. The fifth one is because of the definition of the attack which asks ϵ/n heavy queries, and the sixth one is because $\epsilon = \delta/13 < 1/13$. □

2.4.2 Efficiency of Attack: Proof of Lemma 2.4.2

The proof of attack's efficiency (i.e. Lemma 2.4.2) crucially uses the fact that the attack is *successful*, and uses Lemma 1.3.5.

Recall that Eve's criteria for "heaviness" is based on the distribution $\mathcal{GE}(M, \mathcal{I})$ that is conditioned on her not missing any queries up to this point. However, because we have proven that the event $\mathsf{Good}(M, \mathcal{I})$ has significant probability, queries that are heavy under this distribution are also almost as heavy under the real distribution $\mathcal{E}(M, \mathcal{I})$, and have probability, say, at least $\epsilon/(2n)$. Intuitively this means that, on expectation, Eve will not make more than $O(n^2/\epsilon)$ such queries. The reason is that initially Alice and Bob made at most $2n$ queries that are unknown to Eve, and each query Eve makes decreases the (nonzero) expected number of unknown queries by at least $\epsilon/(2n)$.

We say that a member of the probability space $X \in \mathcal{E}$ is in the event Bad_j, if at the moment that Eve is go to ask her j'th query from the oracle, we have $\Pr_{\mathcal{E}(M,\mathcal{I})}[\neg\mathsf{Good}(M, \mathcal{I})] > 1/2$, where $(M, \mathcal{I})$ are the sequence of messages and Eve's set of query/answer pairs at that moment. Let $\mathsf{Bad} = \bigvee_j \mathsf{Bad}_j$. We define the probability space $\mathcal{E}(\overline{\mathsf{Bad}})$ to be the same as $\mathcal{E}$ with the difference that Eve stops asking more queries whenever Bad_j happens for some j. (The behavior of Alice and

Bob is the same as before.) Note that although $\mathcal{E}$ and $\mathcal{E}(\overline{\mathsf{Bad}})$ are the same probability spaces, because the behavior of Eve is defined differently in them, the events Long and Fail are also defined over them is different ways.[10] However, because $\mathcal{E}$ and $\mathcal{E}(\overline{\mathsf{Bad}})$ are identical as long as Bad does not happen we have $\Pr_{\mathcal{E}}[\mathsf{Bad}] = \Pr_{\mathcal{E}(\overline{\mathsf{Bad}})}[\mathsf{Bad}]$ and more generally for every event D whose definition might depend on Eve's behavior in the system we have $\Pr_{\mathcal{E}}[\mathsf{Bad} \vee D] = \Pr_{\mathcal{E}(\overline{\mathsf{Bad}})}[\mathsf{Bad} \vee D]$.

The proof of efficiency consists of the following two steps.

Step 1. We first use the success property of the attack (i.e., $\Pr_{\mathcal{E}}[\mathsf{Fail}] \leq 3\epsilon/n$) to show that $\Pr_{\mathcal{E}}[\mathsf{Bad}] \leq 6\epsilon$ (Lemma 2.4.7 below) which also means that $\Pr_{\mathcal{E}(\overline{\mathsf{Bad}})}[\mathsf{Bad}] = \Pr_{\mathcal{E}}[\mathsf{Bad}] \leq 6\epsilon$. Note that $\neg\mathsf{Good}(M, \mathcal{I})$ implies that Fail_i has already happened for some i, and so $\neg\mathsf{Good}(M, \mathcal{I})$ implies Fail.

Step 2. We then show that in $\mathcal{E}(\overline{\mathsf{Bad}})$ on average Eve will not ask more than $N = \frac{4n^2}{\epsilon}$ number of queries (see Lemma 2.4.8 below). Since Long is the event that Eve asks more than $\frac{n^2}{\epsilon^2} = \frac{N}{4\epsilon}$ queries, by Markov inequality we have $\Pr_{\mathcal{E}(\overline{\mathsf{Bad}})}[\mathsf{Long}] \leq 4\epsilon$, and therefore we will have

$$\Pr_{\mathcal{E}}[\mathsf{Long}] \leq \Pr_{\mathcal{E}}[\mathsf{Long} \vee \mathsf{Bad}] = \Pr_{\mathcal{E}(\overline{\mathsf{Bad}})}[\mathsf{Long} \vee \mathsf{Bad}] \leq \Pr_{\mathcal{E}(\overline{\mathsf{Bad}})}[\mathsf{Long}] + \Pr_{\mathcal{E}(\overline{\mathsf{Bad}})}[\mathsf{Bad}] \leq 10\epsilon.$$

Now we prove the needed lemmas.

Lemma 2.4.7. $\Pr_{\mathcal{E}}[\mathsf{Bad}] \leq 6\epsilon$.

Proof. We use Lemma 1.3.5 as follows. Let the underlying random variable be $X = \mathcal{E}$, and the event $F = \mathsf{Fail}$. Let the random variable Z_j be the information that Eve learns about X after asking her $(j-1)$'th query, before she asks her j'th query. Namely

[10] In fact, the event Long in $\mathcal{E}(\overline{\mathsf{Bad}})$ is a subset of Long in $\mathcal{E}$, and Fail in $\mathcal{E}$ is a subset of Fail in $\mathcal{E}(\overline{\mathsf{Bad}})$.

$(Z_1, \ldots, Z_j)$ is equal to (M, I) of the moment she wants to ask her j'th query. Let $p = 1/2$, which means B_j is the event that $\Pr[\mathsf{Fail} \mid Z_1, \ldots, Z_j] \geq 1/2$. Lemma 1.3.5 implies that $\Pr[\mathsf{Fail} \mid B] \geq 1/2$.

Note that $\neg\mathsf{Good}(M, \mathcal{I})$ at any moment implies that Fail has already happened, so Bad_j implies B_j and therefore Bad implies B. Now if $\Pr_{\mathcal{E}}[\mathsf{Bad}] \geq 6\epsilon$, we would have $\Pr[\mathsf{Fail}] \geq \Pr[B \wedge \mathsf{Fail}] = \Pr[B] \Pr[\mathsf{Fail} \mid B] \geq \Pr[\mathsf{Bad}](\frac{1}{2}) \geq 3\epsilon$ which contradicts Lemma 2.4.1.

□

Lemma 2.4.8. *Let $\gamma = \frac{\epsilon}{2n}$, and $X = \mathcal{E}(\overline{\mathsf{Bad}})$. If I denotes the set of query/answer pairs that Eve learns by the end of protocol in X, then $\mathbf{E}_X[|I|] \leq \frac{2n}{\gamma} = \frac{4n^2}{\epsilon}$.*

Proof. For a fixed query $q \in \{0,1\}^\ell$, let E_q (resp., F_q) be the event (over X) that Eve (resp., Alice or Bob) asks q. By linearity of expectation we have $\mathbf{E}[|I|] = \sum_q \Pr[E_q]$ and $\sum_q \Pr[F_q] \leq 2n$. We claim that $\Pr[E_q]\gamma \leq \Pr[F_q]$ which would imply the lemma $\mathbf{E}[|I|] = \sum_q \Pr[E_q] \leq \frac{1}{\gamma} \sum_q \Pr[F_q] \leq \frac{2n}{\gamma}$.

To prove $\Pr[E_q]\gamma \leq \Pr[F_q]$, we use Lemma 1.3.5 as follows. The underlying random variable $X = \mathcal{E}(\overline{\mathsf{Bad}})$ (as here), the event $F = F_q$, and the random variable Z_j is as defined in the proof of Lemma 2.4.7. Let $p = \gamma$ which means B_j is the event that $\Pr[F_q \mid Z_1, \ldots, Z_j] \geq \gamma$. Lemma 1.3.5 implies that $\Pr[F_q \mid B] \geq \gamma$.

Note that if Eve asks q from the oracle when she knows $(M, I) = Z_1, \ldots, Z_j$ about X, q has at least ϵ/n probability to be asked by Alice or Bob conditioned on $\mathsf{Good}(M, I)$. But $\Pr[\mathsf{Good}(M, I)] \geq 1/2$ holds in X whenever Eve wants to ask a query, and it means that q is asked by Alice or Bob with probability at least $\frac{\epsilon}{2n} = \gamma$ before. In other words when Eve asks q it holds that $\Pr[F_q \mid Z_1, \ldots, Z_j] \geq \gamma$ which means that the event E_q implies B.

Therefore it holds that $\Pr[F_q] \geq \Pr[F_q \wedge B] = \Pr[B] \Pr[F_q \mid B] \geq \Pr[E_q]\gamma$. □

2.5 Completing the Proof

To complete the proof, we need to **(a)** show how to handle protocols that are not necessarily in normal form and **(b)** show how Eve can recover the secret once she knows the intersection queries. Task **(b)** was already achieved by [98, Theorem 6.2] (although it can be shown that our attack does not need to ask any more queries to find the secret). [98] also showed how one can achieve task **(a)** using a general "compiler" that transforms general protocols to normal form. However that transformation has a quadratic blowup in efficiency that we cannot afford. We show how our attack can be extended to handle general protocols without incurring this cost. (See the full version for the remaining details.)

2.5.1 Removing the Normal Form Assumption

In order to get an attack of the same $(\frac{13n}{\delta})^2$ complexity finding all the intersection queries of Alice and Bob for general form of protocols we do the following.

Attack for Seminormal Protocols

Now we assume that Alice and Bob run a seminormal protocol in which each of them asks at most n number of oracle queries. We prove that the same attack of Section 2.3 finds all the intersection queries with probability $1-\delta$ using $(\frac{13n}{\delta})^2$ number of queries. We only show that the attack is successful in finding all the intersection queries and the same argument as before shows that the efficiency follows from the success property.

Suppose $j \leq O(n^2)$, and it is Bob's turn in the j'th round of the new seminormal protocol. The same argument as before (used for the normal protocols), shows that the probability that Eve misses an intersection query at this point for the first time is

at most $O(\epsilon/n)$, because by not asking any query by Bob, Eve's probability of missing the intersection query can not increase.

Intuitively, we only need to care about the rounds in which Bob does ask a query, but we do not know which n rounds (out of the total $O(n^2)$ rounds) they actually are. One idea is to marginalize the Eve's success probability by fixing the rounds in which Bob asks his queries. But fixing such rounds, would mean that we are fixing Bob's computation to some extent. Therefore we can get the same bound on Eve's failure probability, if we could bound the failure probability in a round, even when Bob's computation till that point is fixed, but we already did so in Lemma 2.4.3. [11]

More formally, let BFail_i be the event that Bob's i'th query is the first intersection query out of $\mathcal{I}$, and similarly let AFail_i be the event that Alice's i'th query is the first intersection query out of $\mathcal{I}$.

Lemma 2.5.1. *For every* $1 \leq i \leq n$, *we have* $\Pr[\mathsf{BFail}_i] \leq \frac{3\epsilon}{2n}$ *and* $\Pr[\mathsf{AFail}_i] \leq \frac{3\epsilon}{2n}$.

Before proving the lemma note that it shows

$$\Pr[\mathsf{Fail}] = \Pr[\bigvee_i (\mathsf{BFail}_i \vee \mathsf{AFail}_i)] \leq \sum_{1 \leq i \leq n} \Pr[\mathsf{BFail}_i] + \Pr[\mathsf{AFail}_i] \leq 3\epsilon.$$

Proof. Let the event BGood_i (in the seminormal protocol) be the event that when Bob asks his i'th query, the intersection queries of Alice and Bob are all inside I (the set of query/answer pairs known to Eve). Let $\mathcal{B}_i$ be the random variable denoting the view of Bob,till he asks his i'th query and let $q(B_i)$ be his i'th query determined by B_i. Also let $(M, I)_i$ be the random variable denoting Eve's information before Bob asks $q(B_i)$.

[11] In fact Lemma 6.2 in [98] is also proved even conditioned on Bob's computation being fixed. Taking this into account, by compiling into seminormal protocols rather than normal ones one can get an $O(n^4)$ attack using [98]'s approach.

So we have $\Pr[\mathsf{BFail}_i] = \sum_{(B_i,M,I)\in \mathrm{Supp}(\mathcal{B}_i\times(M,I)_i)} \Pr_{\mathcal{E}}[B_i, M, I] \Pr_{\mathcal{E}}[\mathsf{BFail}_i \mid B_i, M, I]$. But since BFail_i implies BGood_i we have

$$\Pr_{\mathcal{E}}[\mathsf{BFail}_i \mid B_i, M, I] \leq \Pr_{\mathcal{E}}[\mathsf{BFail}_i \mid B_i, M, I, \mathsf{BGood}_i]$$

and by definition we have $\Pr_{\mathcal{E}}[\mathsf{BFail}_i \mid B_i, M, I, \mathsf{BGood}_i] = \Pr_{\mathcal{GE}(M,I)}[\mathsf{BFail}_i \mid B_i]$. But the proof of Lemma 2.4.3 already showed that $\Pr_{\mathcal{GE}(M,I)}[\mathsf{BFail}_i \mid B_i] \leq \frac{3\epsilon}{2n}$ because it only used the fact that Alice and Bob ask at most n queries each and did not depend on the number of rounds. Hence it holds that

$$\Pr_{\mathcal{E}}[\mathsf{BFail}_i] \leq \sum_{(B_i,M,I)\in \mathrm{Supp}(\mathcal{B}_i\times(M,I)_i)} \Pr_{\mathcal{E}}[B_i, M, I]\frac{3\epsilon}{2n} = \frac{3\epsilon}{2n}.$$

□

Compiling into Normal Form

Suppose i is a round in the original protocol in which Bob is going to ask $k \leq n$ number of queries (k is not known to Eve or Alice) and then send the message m_i to Alice. In the new protocol, this round will be divided into $2n-1$ sub-rounds, which Bob can ask his queries one by one in different rounds, and Alice simply sends $\perp$ messages to Bob. Using $2n-1$ number of rounds rather than $2k-1$ (in case k is known to Alice and Bob at the beginning of round i in the original protocol), is because k might need to be kept secret, and this way the new protocol does not leak any information about k more than the original protocol did.

Now we fix i and describe how the new protocol works in the sub-rounds of this original round. In the j'th sub-round (of round i of the original protocol) if j is even, Alice will just send the message $\perp$ to Bob. So let j be an odd number. If $j \leq k$, Bob

will ask his j'th query which he was going to ask in the i'th round of the original protocol, and if $j > k$ he asks no query. If $j < 2n-1$, Bob sends also the message $\perp$ to Alice in the sub-round j, and if $j = 2n-1$ he sends his message m_i to Alice. It is clear that this artificial change only increases the number of rounds and will not give Eve any extra information, and therefore it is as secure as the original protocol. In the actual attack, Eve will pretend that Alice and Bob are sending the extra $\perp$ messages to each other in the sub-rounds and will attack the protocol in the seminormal form, and as we will prove she finds all the intersection queries with probability $1-\delta$ using $(\frac{13n}{\delta})^2$ number of queries.

2.5.2 Finding the Secret

Now, we turn to the question of finding the secret. Theorem 6.2 in [98] shows that once one finds all the intersection queries, with $O(n^2)$ more queries they can also find the actual secret. Here we use the properties of our attack to show that we can do so even without asking more queries.

Theorem 2.5.2. *Assume that the total number of queries asked by Alice and Bob is at most n each, and their outputs agree with probability at least ρ having access to a random oracle. Then for every $0 < \delta < 1$ there is an adversary Eve asking at most $(\frac{16n}{\delta})^2$ number of queries such that Eve's output agrees with Bob's output with probability at least $\rho - \delta$.*

Proof. Let assume that in the last round of the protocol Alice sends a special message LAST to Bob. In order to find the secret Eve runs the attack of Section 2.3.1 and at the end (when Alice has sent LAST and Eve has asked her queries from the oracle), Eve samples $(A, B) \xleftarrow{R} \mathcal{GE}(M, \mathcal{I})$ (where $(M, \mathcal{I})$ is Eve's information at the moment) and outputs the secret $s(A)$ determined by Alice's view A. Now we prove that her

secret agrees with Bob's secret with probability at least $\rho - 16\epsilon$ and the theorem follows by setting $\delta = \epsilon/16$.

Let the random variables $\mathcal{A}, \mathcal{B}, \mathcal{E}$ be in order the view of Alice, Bob, and Eve at the end of the game. Let $\hat{\mathcal{A}}$ be the random variable generated by sampling $(A, B) \stackrel{R}{\leftarrow} \mathcal{GE}(M, \mathcal{I})$ where $M, \mathcal{I}$ are the information specified in $\mathcal{E}$ and choosing A from it. (So $s(\hat{\mathcal{A}})$ is Eve's output.) We will prove that the statistical distance between $(\mathcal{A}, \mathcal{B}, \mathcal{E})$ and $(\hat{\mathcal{A}}, \mathcal{B}, \mathcal{E})$ is at most 16ϵ which shows that $|\Pr[s(\mathcal{A}) = s(\mathcal{B})] - \Pr[s(\hat{\mathcal{A}}) = s(\mathcal{B})]| \leq 16\epsilon$. For $(A, B, E) \in \mathrm{Supp}(\mathcal{A} \times \mathcal{B} \times \mathcal{E})$ we say the event $\mathsf{Good}(A, B, E)$ holds if A and B do not have any intersection query out of $\mathcal{I}$ where $(M, \mathcal{I}) = E$. The proof follows from the following three claims:

1. $\Pr[\neg\mathsf{Good}(\mathcal{A}, \mathcal{B}, \mathcal{E})] \leq 13\epsilon$.

2. $\Pr[\neg\mathsf{Good}(\hat{\mathcal{A}}, \mathcal{B}, \mathcal{E})] \leq \epsilon$.

3. The statistical distance between

$$(\mathcal{A}, \mathcal{B}, \mathcal{E}) \mid \mathsf{Good}(\mathcal{A}, \mathcal{B}, \mathcal{E}) \text{ and } (\hat{\mathcal{A}}, \mathcal{B}, \mathcal{E}) \mid \mathsf{Good}(\hat{\mathcal{A}}, \mathcal{B}, \mathcal{E})$$

   is at most 2ϵ.

The first claim follows from the proof of Theorem 2.3.1. The second claim is true because after fixing $\mathcal{E} = (M, \mathcal{I})$, the random variable $\hat{\mathcal{A}}$ is independent of $\mathcal{B}$, and if we fix $\mathcal{B} = B$ any query of $Q(B)$ has chance of at most ϵ/n of being in $Q(\hat{\mathcal{A}})$ and there are at most n such queries.

So we only need to prove the third claim which we do even conditioned on fixing $\mathcal{B} = B$ and fixed $\mathcal{E} = E = (M, \mathcal{I})$. The claim follows from Lemma 2.4.5. Let $G = (V_L, V_R, D)$ be the graph characterization of $\mathcal{GE}(M, \mathcal{I})$. Hence we have:

- The distribution of $\mathcal{A}$ in $(\mathcal{A}, B, E) \mid \mathsf{Good}(\mathcal{A}, B, E)$ is the same as choosing $v \in V_R$ such that $B_v = B$ and then choosing a random neighbor of it $u \xleftarrow{R} N(v)$ and getting A_u.

- The distribution of $\hat{\mathcal{A}}$ in $(\hat{\mathcal{A}}, B, E) \mid \mathsf{Good}(\hat{\mathcal{A}}, B, E)$ is the same as choosing $v \in V_R$ such that $B_v = B$ and then choosing a random edge $(u, v') \xleftarrow{R} D$ conditioned on $u \sim v$ and then getting A_u. This is the same as choosing $u \in N(v)$ at random with probability proportional to $d(u)$.

The two above distributions have statistical distance at most 2ϵ even for a fixed v such that $B_v = B$. The first distribution chooses $u \in N(v)$ uniformly at random, but the second distribution chooses $u \in N(v)$ with probabilities proportional to their degrees. But since for every $u \in V_L$, $(1 - 2\epsilon)|V_R| \leq d(u) \leq |V_R|$, (and $\epsilon < 1/10$) one can easily show that the statistical distance is bounded by 2ϵ. $\square$

Chapter 3

Lamport's Signature Scheme is Optimal

3.1 Introduction

Digital signature schemes allow authentication of messages between parties without shared keys. Signature schemes pose an interesting disconnect between the worlds of theoretical and applied cryptography. From a theoretical point of view, it is natural to divide cryptographic tools into those that can be constructed using one-way functions and those that are not known to have such constructions. Signature schemes, along with private key encryption, message authentication codes, pseudorandom generators and functions, belong to the former camp. In contrast, the known constructions of *public key encryption* are based on *structured* problems that are conjectured to be hard (i.e., problems from number theory or the theory of lattices). From a practical point of view, it is more natural to divide the tools according to the *efficiency* of their best known constructions. The division is actually similar, since schemes based on structured problems typically require both more complicated computations and

larger key size, as they often have non-trivial attacks (e.g., because of the performance of the best known factoring algorithms, to get 2^n security based on factorization one needs to use $\widetilde{\Omega}(n^3)$ bit long integers).

Signature schemes are outlier to this rule: even though they can be constructed using one-way functions, applied cryptographers consider them as relatively inefficient since practical constructions are based on structured hard problems, and thus are significantly less efficient than private key encryption, message authentication codes, pseudorandom functions etc... In particular, very high speed applications shun digital signatures in favor of message authentication codes,[1] even though the latter sometime incur a significant cost in keeping shared private keys among the entities involved (e.g., see [138] and the references therein). The reason is that known constructions of such schemes from one-way functions or other unstructured primitives are quite *inefficient*. This problem already arises in *one-time signatures* [112, 119, 140], that are a relaxation of digital signatures offering security only in the case that the attacker observes at most a single valid signature. The best known constructions for this case require $\Omega(k)$ invocations of the one-way function (or even a random oracle) to achieve 2^k security. In contrast, there are known constructions of message authentication codes, private key encryptions, and pseudorandom generators and functions that use only $O(1)$ queries to a random oracle.

In this chapter, we study the question of whether there exist more efficient constructions of signature schemes from symmetric primitives such as hash functions and block ciphers. We show to a certain extent that the inefficiency of the known constructions is *inherent*.

[1] In contrast to digital signatures that have a public verification key and secret signing key, *message authentication codes* have a single key for both verification and signing, and hence that key must be kept private to maintain security.

3.1.1 Our Results

We consider the efficiency of constructions of one-time signatures using black boxes / oracles that model ideal symmetric primitives: the random oracle, the random permutation oracle, and the ideal cipher oracle (see Section 3.3 for definitions). We wish to study the security of such constructions as a function of the number of queries made to the oracle by the construction (i.e., by the generation, signing, and verification algorithms). Of course, we believe that one-time signatures exist and so there are in fact signature schemes achieving super-polynomial security without making any query to the oracle. Hence we restrict ourselves to bounding the *black-box* security of such schemes. We say that a cryptographic scheme using oracle $\mathcal{O}$ has *black-box security* S if for every $1 \leq T \leq S$, a (potentially computationally unbounded) adversary that makes at most T queries to $\mathcal{O}$ cannot break the scheme with probability larger than T/S (see Definition 3.3.2). Our main result is the following:

Theorem 3.1.1. *Any one-time signature scheme for n-bit messages using at most $q \leq n$ queries to a random oracle has black-box security at most $2^{(1+o(1))q}$ where $o(1)$ goes to zero with q.*

This is in contrast to other primitives such as message authentication codes, collision resistant hash functions, private-key encryption, and pseudorandom functions, that can all be implemented using one or two queries to a random oracle with black-box security that depends exponentially on the length of these queries. We note that Theorem 3.1.1 is tight up to a constant factor in the number of queries, since a simple modification of Lamport's scheme [112] yields $2^{(\alpha-o(1))q}$ black-box security, where $\alpha \sim 0.812$ is equal to $H(c)/(1+c)$, where H is the Shannon entropy function and $c = (3-\sqrt{5})/2$ (see Section 3.5). We also prove several extensions of the main result:

Other oracles. Since our goal is to find out whether signatures can be efficiently constructed from symmetric primitives, it makes sense to study also other primitives than the random oracle. Theorem 3.1.1 extends (with a loss of a constant factor in the number of queries) to the *ideal cipher oracle* and *random permutation oracle* that are also sometimes used to model the idealized security of symmetric primitives such as block ciphers and one-way permutations.

Implementing adversary in $\mathbf{BPP^{NP}}$. The proof of Theorem 3.1.1 shows that for every q-query one-time signature scheme for $\{0,1\}^n$ from random oracle, there is an adversary that breaks it with probability close to 1 using at most $\mathrm{poly}(q)2^q$ queries. However, the *running time* of this adversary can be higher than that. This is inherent, as otherwise we would be proving unconditionally the non-existence of one-time signature schemes. However, we show that this adversary can be implemented in probabilistic polynomial-time using an oracle to an **NP**-complete problem. Thus, similar to what Impagliazzo and Rudich [98] showed for key-exchange, if there were a more efficient construction of signature schemes from random oracles with a proof of security relying on the adversary's efficiency, then this is also a proof that $\mathbf{P} \neq \mathbf{NP}$.

Imperfect completeness. While the standard definition of signature schemes requires the verifier to accept valid signatures with probability 1, one can also consider relaxed variants where the verifier has some small positive probability of rejecting even valid signatures. We say that such signature schemes satisfy *imperfect completeness*. We can extend Theorem 3.1.1 to this case, though to get an attack succeeding with high probability we lose a quadratic factor in the number of queries.

Efficiency of the verifier. Because the signing and the verification algorithms are executed more often than the key generation algorithm, it makes sense to study their efficiency separately rather than just studying the total number of queries. Although in the construction for signature schemes that we will see later (see Section 3.5), the signing algorithm asks only one oracle query and the total number of queries is optimal up to a constant factor, the question about the efficiency of the verifier still remains. We show that (keeping the number of signing queries fixed to one) there is a tradeoff between the number of queries asked by the verification algorithm and the total number of queries, conditioned on getting certain black-box security.

Black-box constructions. As mentioned above, all the symmetric primitives can be constructed from random oracle, random permutation oracle, or ideal cipher oracle by only $O(1)$ queries and get exponential security over the length of the queries. Therefore, our lower-bounds on signatures from ideal oracles yield as corollaries lower-bounds on the efficiency of signatures from symmetric primitives when the construction is black box. This holds even when the one-way permutation used in the construction has $n/2$ hardcore bits. The latter answers a question raised by [61]. Our results reject the existence of black-box constructions unconditionally (similar to [82], while the results of [61] show the existence of one-way function as a consequence. We prove the strongest possible form of lower-bound on the efficiency of black box constructions of signatures from symmetric primitives. Namely, we show that black-box constructions of signature schemes for n-bit messages based on exponentially hard symmetric primitives of security parameter n, need to make at least $\Omega(n)$ calls to the primitive.

Note on the random oracle model. Although the *random oracle model* [13] (and its cousin the *ideal cipher model*) is frequently used as an idealization of the properties enjoyed by certain constructions such as the SHA-1 hash function [126] and the AES block cipher [43], it has drawn a lot of criticism as this idealization is not generally justified [38]. However, for the sake of *lower-bounds* (as is our concern here) this idealization seems appropriate, as it is a clean way to encapsulate all the attractive properties that could be obtained by constructions such as SHA-1,AES, etc..

Taxonomy of black-box reductions. Reingold, Trevisan and Vadhan [143] study various notions of "black-boxness" of security proofs in cryptography according to whether a construction of a cryptographic tool based on an underlying primitive uses this primitive as a black box, and whether its security proof uses the adversary as a black box. Those definitions are not in the oracle model that we are concerned here. They call a construction for primitive A from primitive B black-box, if the implementation of A uses B as a black box. The security reduction which converts an adversary for the implementation of A to an adversary for B could have different levels of being black box [2]. However, in the oracle based constructions studied here, the implementation reduction is always forced to be black-box, and for the proof of security, there is no security measure defined for the primitive used (i.e. the oracle) to which we could reduce the security of our construction. One common way to prove security for oracle based constructions is to rely on the statistical properties of the oracle and show that any (even computationally unbounded) adversary breaking the implementation needs to ask *many* queries from the oracle. This gives a quantitative

[2]It could be fully black-box, semi black-box, or non-black-box, and if the implementation reduction is black box, the whole construction is called, (resp.,) fully black-box, semi black-box, or weakly black-box.

security guarantee and is called a *black-box* proof of security in the oracle model. A *non-black-box* proof of security in this model, is a proof showing that any adversary who runs in time poly(n, T) where n is the input length and T the number of oracle queries it asks, needs to ask many queries from the oracle. In this work, we give a lower-bound on the number of queries needed to get *black-box* security S for one-time signatures in various ideal oracle models, and also show that if $\mathbf{P} = \mathbf{NP}$, then this bound holds for *non-black-box* proofs of security as well. We note that if one-way functions exist, then there do exist constructions making no query to the random oracle with super-polynomial non-black-box security. As we mentioned before, our lower-bounds in the ideal oracle models yield some lower-bounds on the efficiency of one-time signatures from symmetric primitives in the standard model of [143]. We also note that there do exist cryptographic constructions that use the primitive [70, 71] or the adversary [12] in a non-black-box way, but at the moment all of the known highly efficient cryptographic constructions (e.g., those used in practice) are black box, in the sense that if they use a generic underlying primitive (i.e., not based on specific problems such as factoring) then it's used as a black-box and if they have a proof of security then the proof treats the adversary as a black box.

3.1.2 Prior Work

To the best of our knowledge, this is the first lower-bound on the number of random oracle queries needed to construct signature schemes. Starting with the seminal paper of Impagliazzo and Rudich [98], that showed that there is no construction of a key exchange protocol from a random oracle with super-polynomial black-box security, and therefore rejecting black-box constructions of key exchange protocols from one-way function, several works have investigated the *existence* of black-box constructions reducing one kind of cryptographic scheme to another. However, only few works

studied the *efficiency* of such constructions [61, 109]. Of these, the most relevant is the paper by Gennaro, Gertner, Katz, and Trevisan [61]. They considered the efficiency of basing various cryptographic primitives on one-way permutations (OWP) secure against S-sized circuits, and proved that to achieve super-polynomial security **(1)** pseudorandom generators with ℓ bits of stretch require $\Omega(\ell/\log S)$ invocations of the OWP, **(2)** universal one-way hash functions compressing their input by ℓ bits require $\Omega(\ell/\log S)$ invocations, **(3)** private key encryption schemes for messages of length n with key length k require $\Omega((n-k)/\log S)$ invocations, and (most relevant for us) **(4)** one-time signature schemes for n-bit messages require $\Omega(n/\log S)$ invocations.[3]

However, the one-way permutation oracle used by [61] was very far from being a random oracle.[4] Indeed, the applications **(1)**, **(2)**, and **(3)** can be implemented using only a constant number of calls to a random oracle, and correspondingly are considered to have efficient practical implementations. Thus, [61] did not answer the question of whether signature schemes can be efficiently constructed from efficient symmetric key primitives such as hash functions and block ciphers. It is this question that we are concerned with in this chapter. Thus, on a technical level our work is quite different from [61] (as we work with a random oracle and cannot "tamper" with it to prove our lower-bound) and in fact is more similar to the techniques in the original work of Impagliazzo and Rudich [98]. We note that this work partially answers a question of [61], as it implies that any black-box construction of one-time signatures from one-way permutation $p : \{0,1\}^n \mapsto \{0,1\}^n$ with even $n/2$ hard-core bits requires at least $\Omega(n)$ queries to the permutation.

[3] Otherwise, we can construct a one-way function directly.

[4] They considered an oracle that applies a random permutation on the first t bits of its n-bit input, for $t \ll n$, and leaves the rest of the $n-t$ bits unchanged. This is a one-way permutation with $2^{\Omega(t)}$ security.

Several works [19, 20, 53, 119, 155] considered generalizations of Lamport's one-time signature scheme. Some of these achieve shorter keys and signatures, although their relation between the number of queries and security (up to a constant factor) is at most a constant factor better than Lamport's scheme (as we show is inherent).

3.2 Our Techniques

We now give a high level overview of the ideas behind the proof of Theorem 3.1.1. Our description ignores several subtle issues, and the reader is referred to Section 3.4 for the full proof. To understand the proof of the lower-bound,[5] it is instructive to review the known *upper-bounds* and in particular the simple one-time signature scheme of Lamport [112]. To sign messages of length n with security parameter ℓ using a random oracle $\mathcal{O}$ (that we model as a random function from $\{0,1\}^\ell$ to $\{0,1\}^\ell$) the scheme works as follows:

- Generate the public verification key vk by choosing $2n$ random strings $\{x_i^b\}_{i\in[n],b\in\{0,1\}}$ in $\{0,1\}^\ell$ and setting vk to be the sequence $\{y_i^b\}_{i\in[n],b\in\{0,1\}}$ for $y_i^b = \mathcal{O}(x_i^b)$.
- To sign a message $\alpha \in \{0,1\}^n$, simply reveal the preimages in the set $\{x_i^b\}_{i\in[n],b\in\{0,1\}}$ that correspond to the bits of α. That is, the signature is $x_1^{\alpha_1}, \ldots, x_n^{\alpha_n}$.
- The verifier checks that indeed $\mathcal{O}(x_i^{\alpha_i}) = y_i^{\alpha_i}$ for every $i \in [n]$.

This scheme uses $3n$ queries. It can be shown that it has $2^{\Omega(\ell)}$ security. Note that in this case the security can be arbitrarily large *independently* of the number of queries. Indeed, note that Theorem 3.1.1 requires that the number of queries q is not larger than the length of the messages to be signed. Lamport's scheme can be easily

[5] We use the terms "lower-bound" and "upper-bound" in their traditional crypto/complexity meaning of negative results vs. positive results. Of course one can view Theorem 3.1.1 as either upper-bounding the security or lower-bounding the number of queries.

modified to work for unbounded size messages by following the well known "hash-and-sign" paradigm: first use the random oracle to hash the message to length k, and then apply Lamport's scheme to the hashed value. This will result in a scheme with $3k + 2$ queries and (by the birthday bound) $2^{k/2}$ black-box security (see Section 3.5 for some improvements). We see that now indeed the security is bounded by $2^{O(q)}$ (where $q = 3k + 2$ is the number of queries), regardless of the length ℓ of the queries.

The above discussion shows that to prove Theorem 3.1.1, we will need to use the fact that there is a large number of potential messages, which is indeed what we do. Note that the reason that the hash-and-sign variant of Lamport's scheme only achieves $2^{k/2}$ security is that if a pair of messages α, β satisfies $\mathcal{O}_k(\alpha) = \mathcal{O}_k(\beta)$ (where $\mathcal{O}_k(x)$ denotes the first k bits of $\mathcal{O}(x)$), then they have the same signature, and so a signature for α allows an adversary to forge a signature on β. We will try to generalize this observation to arbitrary signature schemes. For every such scheme $\mathcal{S}$ and two messages α, β (after fixing the oracle and the randomness of the system), we will say that "α is useful for β" if they satisfy a certain condition. Then (roughly speaking) we will prove that: **(A)** if α is useful for β then a signature on α can be used to compute a signature on β by asking at most $2^{O(q)}$ oracle queries (where q is the total number of queries made by the scheme $\mathcal{S}$), and **(B)** if α and β are chosen at random from a large enough space of messages, then α will be useful for β with probability at least $2^{-O(q)}$. Together **(A)** and **(B)** imply that, as long as the space of possible messages is large enough, then the black-box security of $\mathcal{S}$ is bounded by $2^{O(q)}$, since the adversary can find a useful pair of messages α, β with probability 2^{-q}, ask for a signature on α and use that to forge a signature on β by asking 2^q queries.[6]

[6]The actual adversary we'll show will operate by asking $\text{poly}(q)2^q$ queries, and it succeeds with probability almost 1, see the proof of Theorem 3.4.1.

3.2.1 Defining the Usefulness Condition

This proof strategy rests of course on the ability to find an appropriate condition "α is useful for β" for every one-time signature scheme $\mathcal{S}$. This is what we describe now. For now, we will assume that only the key generation algorithm of $\mathcal{S}$ is probabilistic, and that both the signing and verification algorithms are deterministic.[7] For every fixed randomness for the generation algorithm, fixed oracle, and a message α, we define G, S_α and V_α to be the sets of queries (resp.,) made by the generation, signing, and verification algorithms where the last two are applied on the message α.

First attempt. Observe that in the hash-and-sign variant of Lamport's scheme, α and β have the same signature if $V_\alpha = V_\beta$. This motivates stipulating for every signature scheme that α is useful for β if $V_\beta \subseteq V_\alpha$. This definition satisfies Property **(A)** above: if we know all the queries that the verifier will make on a signature of β, then finding a signature that makes it accept can be done by an exponential-time exhaustive search that does not make any oracle queries at all. The problem is that it might not satisfy **(B)**: it's easy to make the verifier ask, when verifying a signature for α, a query that uniquely depends on α, thus ensuring $V_\beta \nsubseteq V_\alpha$ for every distinct α, β.

Second attempt. A natural intuition is that verifier queries that do not correspond to queries made by the generation algorithm are sort of "irrelevant"— after all, in Lamport's scheme all the queries the verifier makes are a subset of the queries made by the generation algorithm. Thus, we might try to define that α is useful for β if $V_\beta \cap G \subseteq V_\alpha$. Since G has at most q queries, and so at most 2^q subsets, this definition

[7] We study the randomized verifier in Section 3.6.1, but assuming that the signer is deterministic is without loss of generality. That is because the key generator can give, through the secret key, a secret seed s to the signer, and the signer would use $\mathcal{O}(s, \alpha)$ as the randomness needed to sign the message α.

satisfies Property **(B)** since if α and β are randomly chosen from a set of size 2^q then α will be useful for β with probability at least 2^{-2q}. Unfortunately, it does not satisfy Property **(A)**: there is a signature scheme for which every pair of messages α, β satisfies this condition even when a signature for α cannot be used to forge a signature on β.[8]

Our actual condition. The condition we actually use, roughly speaking, is that α is useful for β if

$$V_\beta \cap (G \cup S_\alpha) \subseteq V_\alpha \quad . \tag{3.2.1}$$

Using Bollobás's Inequality [29] (see the proof of Claim 3.4.7) it can be shown that the condition (3.2.1) satisfies Property **(B)**. It's less obvious why it satisfies Property **(A)**— to see this we need to see how our adversary will operate. The high level description of our attack is as follows:

1. **Input: key generation.** The adversary receives the verification key vk.

2. **Request signature.** Choose $\alpha \neq \beta \xleftarrow{R} \{0,1\}^n$ at random, and get σ_α, the signature of α.

3. **Learning oracle queries.** Run $\mathrm{Ver}(\mathsf{vk}, \alpha, \sigma_\alpha)$ to learn the set V_α of oracle queries that it asks and their answers. (Later we will modify this step somewhat, and ask some more oracle queries.)

4. **Sampling a possible transcript.** Conditioned on knowing $\mathsf{vk}, \sigma_\alpha$, and answers of V_α, *guess*: the value of sk, the sets G and S_α, and their answers. Let $\widetilde{\mathsf{sk}}$, $\widetilde{G}$, and $\widetilde{S}_\alpha$ be the guesses.

[8] Such an example can be obtained by the variant of Lamport's scheme where each signer uses the verification key vk to sign a new verification key vk' (the randomness for which is part of the secret key), and then signs the message using the secret key corresponding to vk'. In this case $V_\alpha \cap G = V_\beta \cap G$ for every pair α, β, even if a signature on α cannot be used to compute a signature on β.

5. **Forging.** Sign the message β by using $\widetilde{\mathsf{sk}}$ and sticking to the oracle answers guessed for queries in $\widetilde{G} \cup \widetilde{S}_\alpha$ to get σ_β. That is, if we wanted to ask a an oracle query in $\widetilde{G} \cup \widetilde{S}_\alpha$, use the guessed answer, and otherwise ask the real oracle $\mathcal{O}$. Output σ_β.

Note that the queries for which we might have guessed a wrong answer are in the set $(\widetilde{G} \cup \widetilde{S}_\alpha) \backslash V_\alpha$, because we did the guesses conditioned on knowing V_α and its answers. Suppose that during the verification of (β, σ_β), none of these queries is asked from the oracle (i.e. $V_\beta \cap (\widetilde{G} \cup \widetilde{S}_\alpha) \subset V_\alpha$). Then we can *pretend* that our guesses were correct. That is, because the answers to different queries of random oracle are independent, as far as the verifier is concerned our guesses could be right, and hence by definition, the verification of (β, σ_β) must accept with probability 1.

The description of the attack above shows that a similar condition to the condition (3.2.1), namely

$$V_\beta \cap (\widetilde{G} \cup \widetilde{S}_\alpha) \subset V_\alpha \ , \tag{3.2.2}$$

has Property **(A)**. But condition (3.2.2) might not have Property **(B)**. We cope with this by ensuring that the attacker has sufficient information so that (essentially) whenever (3.2.1) happens, (3.2.2) also happens. This is accomplished by learning more oracle queries before making the guesses. Namely, we learn all the queries that are in the set $\widetilde{G} \cup \widetilde{S}_\alpha$ with some noticeable probability (conditioned on what we know about them). We then use a careful hybrid argument (that involves the most technical part of the proof) to show that after performing this learning, the condition (3.2.2) occurs with probability at least as large as the probability that (3.2.2) occurs (up to some lower order terms). Thus our actual usefulness condition will be (3.2.2), though for the complete definition of the sets $\widetilde{G}, \widetilde{S}_\alpha$ involved in it, one needs to go into the details of the proof of Theorem 3.4.1).

3.3 Signature Schemes in Oracle Models

We define the notion of one-time signature schemes and their black-box security. We specialize our definition to the case that the signature schemes use an oracle $\mathcal{O}$ that may also be chosen from some probability distribution. We use the standard notation $A^{\mathcal{O}}(x)$ to denote the output of an algorithm A on input x with access to oracle $\mathcal{O}$.

Definition 3.3.1. An oracle *signature scheme* (with perfect completeness) for n bit messages is a triple of oracle algorithms $(\mathrm{Gen}, \mathrm{Sign}, \mathrm{Ver})$ (where Gen could be probabilistic) with the following property: for every oracle $\mathcal{O}$, if $(\mathsf{sk}, \mathsf{vk})$ is a pair that is output by $\mathrm{Gen}^{\mathcal{O}}(1^n)$ with positive probability, then for every $\alpha \in \{0,1\}^n$, $\mathrm{Ver}^{\mathcal{O}}(\mathsf{vk}, \alpha, \mathrm{Sign}^{\mathcal{O}}(\mathsf{sk}, \alpha)) = 1$. We call sk the *signing key* and vk the *verification key*.

One can also make a relaxed requirement that the verification algorithm only needs to accept valid signatures with probability 0.9 (where this probability is over the verifier's coins only). We say that such relaxed signature schemes have *imperfect completeness*, and we will consider such schemes in Section 3.6.1. If the oracle algorithms of the Definition 3.3.1 run in polynomial-time, then we call the signature scheme *efficient*. Note that we consider (not necessarily efficient) signature algorithms on a finite set of messages. For upper-bounds (i.e., positive results) one would want uniform efficient algorithms that could handle any size of message, but for a lower-bound (i.e., a negative result), this simpler definition will do.

So far, we did not say anything about the security. In the following definition we specify the "game" in which the adversary participates and tries to break the system and give a quantitative measure for the security.

Definition 3.3.2. For every $S \in \mathbb{N}$, the oracle signature scheme $(\mathrm{Gen}, \mathrm{Sign}, \mathrm{Ver})$ is a *one-time* signature scheme with *black-box security* S, if for every message $\alpha \in \{0,1\}^n$, $1 \leq T \leq S$, and adversary algorithm A that makes at most T queries to its

oracle, $\Pr[\mathrm{Ver}(\mathsf{vk}, \alpha^*, \sigma^*) = 1 \text{ where } (\alpha^*, \sigma^*) = A^{\mathcal{O}}(\mathsf{vk}, \mathrm{Sign}^{\mathcal{O}}(\mathsf{sk}, \alpha)) \text{ and } \alpha^* \neq \alpha] \leq \frac{T}{S}$, where $(\mathsf{sk}, \mathsf{vk}) = \mathrm{Gen}^{\mathcal{O}}(1^n)$, and this probability is over the coins of all algorithms (Gen, Sign, Ver, and A), and the choice of the oracle $\mathcal{O}$.

This is a slightly weaker definition of security than the standard definition, since we are not allowing the adversary to choose the message α based on the public key. However, this is again fine for lower-bounds (the known upper-bounds do satisfy the stronger definition). Also, some texts use $1/S$ (rather than T/S) as the bound on the success probability. Security according to either one of these definitions is always at most quadratically related, but we feel Definition 3.3.2 is more precise.

In a *non-black-box* proof of security, the running time of the adversary is utilized in order to prove the security of the system:

Definition 3.3.3. For every $S \in \mathbb{N}$, the oracle signature scheme (Gen, Sign, Ver) is a one-time signature scheme with *non-black-box* security S, if for every message $\alpha \in \{0,1\}^n$, $T \leq S$, and adversary algorithm A_T that makes at most T oracle queries and runs in time $\mathrm{poly}(n, T)$, $\Pr[\mathrm{Ver}(\mathsf{vk}, \alpha^*, \sigma^*) = 1 \text{ where } (\alpha^*, \sigma^*) = A_T^{\mathcal{O}}(\mathsf{vk}, \mathrm{Sign}^{\mathcal{O}}(\mathsf{sk}, \alpha)) \text{ and } \alpha^* \neq \alpha] \leq \frac{T}{S}$, where $(\mathsf{sk}, \mathsf{vk}) = \mathrm{Gen}^{\mathcal{O}}(1^n)$, and this probability is over the coins of all algorithms (Gen, Sign, Ver, and A_T), and the choice of the oracle $\mathcal{O}$.

Oracles. In this work, as for the oracle signature schemes, we only use one of the following oracles: **(1)** The *random oracle* returns on input $x \in \{0,1\}^n$ the value $f(x)$ where f is a random function from $\{0,1\}^n$ to $\{0,1\}^n$.[9] **(2)** The *random permutation oracle* returns on input $x \in \{0,1\}^n$ the value $f(x)$ where f is a random permutation on $\{0,1\}^n$. **(3)** The *ideal cipher oracle* with message length n, returns on input

[9]More generally, f can be a function from n to $\ell(n)$ for some function $\ell : \mathbb{N} \mapsto \mathbb{N}$, but using standard padding arguments we may assume $\ell(n) = n$.

(k, x, d) where $k \in \{0,1\}^*$, $x \in \{0,1\}^n$ and $d \in \{\mathsf{F}, \mathsf{B}\}$, $f_k(x)$ if $d = \mathsf{F}$ and $f_k^{-1}(x)$ if $d = \mathsf{B}$, where for every $k \in \{0,1\}^*$, f_k is a random permutation on $\{0,1\}^n$. These three oracles are standard idealizations of (respectively) hash functions, one-way permutations, and block ciphers (see also Section 3.7).

3.4 Proof of the Main Result

Theorem 3.4.1. *Let* $(\mathrm{Gen}, \mathrm{Sign}, \mathrm{Ver})$ *be a one-time oracle signature scheme (with perfect completeness) in random oracle model for the space of messages* $\mathcal{M}$ *in which the total number of oracle queries asked by* $\mathrm{Gen}, \mathrm{Sign}$*, and* Ver *is at most* q*, and* $|\mathcal{M}| \geq \frac{\binom{q}{q/2}}{\lambda}$*. Then there is a (computationally unbounded) adversary which asks at most* $O(\frac{q^2\binom{q}{q/2}}{\lambda\delta^2}) = O(\frac{q^{1.5}2^q}{\lambda\delta^2})$ *oracle queries and breaks the scheme with probability* $1 - (\lambda + \delta)$*. This probability is over the randomness of the oracle as well as the coin tosses of the key generation algorithm and the adversary.*

Theorem 3.4.1 implies Theorem 3.1.1 via the following corollary:

Corollary 3.4.2. *Let* $(\mathrm{Gen}, \mathrm{Sign}, \mathrm{Ver})$ *be a one-time oracle signature for the messages* $\mathcal{M} = \{0,1\}^n$ *in the random oracle model in which the total queries asked by the scheme is at most* q *where* $q \leq n$*, then there is an adversary asking* $2^{(1+o(1))q}$ *queries breaking the scheme with probability at least* $1 - o(1)$ *and at least* 0.49 *for any* $q \geq 1$*.*

Proof. Let $\delta = \lambda = \binom{q}{q/2}/2^q = \theta(q^{-1/2}) = o(1)$, so we have $|\mathcal{M}| = 2^n \geq 2^q = \binom{q}{q/2}/\lambda$. Therefore we get an adversary asking $O(q^{3.5}\binom{q}{q/2}) = O(q^3 2^q) = 2^{(1+o(1))q}$ queries breaking the scheme with probability $1 - o(1)$. Thus the black-box security of the scheme is at most by $\frac{2^{(1+o(1))q}}{1-o(1)} = 2^{(1+o(1))q}$. For any $q \geq 1$, λ can be as small as $\binom{1}{0}/2^1 = 1/2$, and by taking $\delta = 0.01$ the success probability will be at least 0.49. $\square$

We now turn to proving Theorem 3.4.1. Let $(\mathrm{Gen}, \mathrm{Sign}, \mathrm{Ver})$ be as in the theorem's statement. We assume that only Gen is probabilistic, and Sign and Ver are deterministic. We also assume that all the oracle queries are of length ℓ. Since we assume the signature has perfect completeness, these assumptions can be easily shown to be without loss of generality. (In the case of imperfect completeness the verifier algorithm is inherently probabilistic; this case is studied in Section 3.6.1.) We will show an adversary that breaks the signature system with probability $1 - (\lambda + O(\delta))$, which implies Theorem 3.4.1 by simply changing δ to δ/c for some constant c.

The Adversary's Algorithm

Our adversary Adv will operate as follows:

Input: Key generation. The adversary receives a verification key vk, where $(\mathsf{vk}, \mathsf{sk}) = \mathrm{Gen}(1^n)$.

Step 1: Request signature. Let $\beta_0, \ldots, \beta_{N-1}$ denote the first $N = \frac{\binom{q}{q/2}}{\lambda}$ distinct messages (in lexicographic order) in $\mathcal{M}$. Let $\alpha_0, \ldots, \alpha_{N-1}$ be a random permutation of $\beta_0, \ldots, \beta_{N-1}$. Adv asks for a signature on α_0 and verifies it (note that α_0 is chosen independently of the public key). We denote the obtained signature by σ_0, and we denote by T_0 the *transcript* of the algorithms run so far, which includes the random tape of the key generation algorithm, all the queries made by the key generation, signing, and verification algorithms, and the answers to these queries. So T_0 completely describes the running of the algorithms so far. (Note that Adv only has partial information on T_0.)

Step 2: Learning query/answer pairs. We denote by L_0 the information that Adv currently has on the oracle $\mathcal{O}$ and the randomness of the generation algorithm: that is, L_0 consists of vk, σ_0 and the queries made by the verifying

algorithm Ver on input vk, σ_0, along with the answers to these queries. Let $\epsilon = \frac{\delta}{qN}$, and $M = \frac{q}{\epsilon\delta} = \frac{q^2N}{\delta^2}$. For $i = 1, \ldots, M$, do the following:

1. Let $\mathbf{D}_{i-1}$ be the distribution of T_0, the transcript of the first step, conditioned on only knowing L_{i-1}.
2. We let $Q(L_{i-1})$ denote the queries appearing in L_{i-1}. If there exists a string $x \in \{0,1\}^\ell \setminus Q(L_{i-1})$ that is queried with probability at least ϵ in $\mathbf{D}_i$, then Adv lets L_i be L_{i-1} concatenated with the query/answer pair $(x_i, \mathcal{O}(x_i))$, where x_i is the lexicographically first such string. Otherwise, $L_i = L_{i-1}$.

Step 3: Sampling a possible transcript. Adv generates a random transcript $\widetilde{T}_0$ according to the distribution $\mathbf{D}_M$. Note that $\widetilde{T}_0$ also determines a secret signing key, which we denote by $\widetilde{\mathsf{sk}}$ ($\widetilde{\mathsf{sk}}$ may or may not equal the "true" signing key sk). $\widetilde{T}_0$ may also determine some query/answer pairs that were not in L_M, and hence may not agree with the the actual answers of the "true" oracle $\mathcal{O}$. We denote by $\widetilde{\mathcal{O}}$ the oracle that on input x, if x appears as a query in $\widetilde{T}_0$ then $\widetilde{\mathcal{O}}(x)$ outputs the corresponding answer, and otherwise $\widetilde{\mathcal{O}}(x) = \mathcal{O}(x)$.

Step 4: Forging. For every $j = 1, \ldots, N-1$, Adv uses $\widetilde{\mathsf{sk}}$ and the oracle $\widetilde{\mathcal{O}}$ to compute a signature on the message α_j, which it then tries to verify this time using vk and the "true" oracle $\mathcal{O}$. Adv outputs the first signature that passes verification.

Analysis

The number of queries asked during the attack is at most $M + qN = \frac{q^2N}{\delta^2} + qN \leq \frac{2q^2N}{\delta^2} = O(\frac{q^2\binom{q}{q/2}}{\lambda\delta^2})$. To analyze the success probability of Adv we will prove the following lemma:

Lemma 3.4.3. *For every $j \in [0..N-1]$, let V_j denote the set of queries made by* Adv *when verifying the signature on α_j. Let $\widetilde{G}$ and $\widetilde{S}_0$ be the sets of queries made by the generation and signing algorithms according to the transcript $\widetilde{T}_0$. For every $j \geq 1$, let E_j be the event that $V_j \cap (\widetilde{G} \cup \widetilde{S}_0) \subseteq V_0$. Then,*

$$\Pr[\cup_{j\in[1..N-1]} E_j] = 1 - (\lambda + 2\delta)$$

Note that the event E_j corresponds to the condition that "α_0 is useful for α_j" described in Section 3.2. Lemma 3.4.3 implies Theorem 3.4.1 since if the event E_j holds then when verifying the signature for α_j, the verifier never asks a query on which the oracles $\mathcal{O}$ and $\widetilde{\mathcal{O}}$ differ (these oracles can differ only on queries in $(\widetilde{G} \cup \widetilde{S}_0) \setminus V_0$). But if the verifier uses the same oracle $\widetilde{\mathcal{O}}$ used by the generation and signing algorithm, then by the definition of a signature scheme, it must accept the signature.

3.4.1 Proof of Lemma 3.4.3

It turns out that using known combinatorial techniques, one can show that $\cup_j E_j$ holds with high probability if all signatures and verifications were to use the "true" oracle $\mathcal{O}$ and signing key sk (as opposed to $\widetilde{O}$ and $\widetilde{\mathsf{sk}}$). The idea behind the proof is to show this holds in our case using a hybrid argument. Specifically, we define four distributions $\mathsf{Hyb}^0, \mathsf{Hyb}^1, \mathsf{Hyb}^2, \mathsf{Hyb}^3$, where Hyb^0 corresponds to $\widetilde{T}_0$ joint with all the oracle queries/answers that the adversary gets during the signing and verification algorithms on α_j for $j \geq 1$ (we call this information the *transcript* of the experiment), and Hyb^3 corresponds to T_0 (the real transcript of the first step) joint with the rest of the system's transcript if we use the "true" oracle and signing key (so the adversary is not doing anything in generating Hyb^3). We will prove the lemma by showing that the probability of $\cup_j E_j$ is almost the same in all these four distributions.

Definition of hybrid distributions. The four hybrid distributions $\mathsf{Hyb}^0, .., \mathsf{Hyb}^3$ are defined as follows:

Hyb^0**:** This is the distribution of $\widetilde{T}_0, T_1, \ldots, T_{N-1}$, where $\widetilde{T}_0$ denotes the transcript sampled by Adv in Step 3, while T_j (for $j \geq 1$) denotes the transcript of the j'th signature (i.e., the queries and answers of the signing and verification algorithms on α_j) as generated by Adv in Step 4. Note that T_0 and $\widetilde{T}_0$ describe also the running of the key generation while T_j for $j \geq 1$ do not.

Hyb^1**:** This is the same distribution as Hyb^0, except that now in Step 4 of the attack, the adversary uses the modified oracle $\widetilde{\mathcal{O}}$ for both signing and verifying the signatures on $\alpha_1, \ldots, \alpha_{N-1}$ (recall that in Hyb^0 the oracle $\widetilde{\mathcal{O}}$ is only used for signing).

Hyb^2**:** This is the same distribution as Hyb^1, except that we make a slight modification in the definition of $\widetilde{\mathcal{O}}$: for every query x that was asked by the generation, signing, and verification algorithms in the Input step and Step 1 (i.e., for every query in T_0), we answer with $\mathcal{O}(x)$ *only* if x also appears in L_M. Otherwise, we answer this query with a completely random value. Note that all the queries of the verification are in L_0 and so in L_M as well. In other words, $\widetilde{\mathcal{O}}$ agrees with $\mathcal{O}$ on all the queries that Adv has asked from $\mathcal{O}$ till the end of Step 2, and all the others are answered completely at random.

Hyb^3**:** This is the same distribution as the previous ones, with the difference that $\widetilde{T}_0$ is chosen equal to T_0 (and so, there is no point in neither Step 2 of the attack nor defining $\widetilde{\mathcal{O}}$ anymore). In other words, this is the transcript (randomness and all query/answer pairs) of the following experiment: **(1)** Generate signing and verification keys $(\mathsf{sk}, \mathsf{vk})$ using a random oracle $\mathcal{O}$ **(2)** for $j = 0 \ldots N-1$, sign α_j and verify the signature using sk, vk and $\mathcal{O}$.

Note that the hybrid distributions Hyb^i are over the coin tosses of the oracle, the key generation algorithm, and the adversary. Lemma 3.4.3 follows immediately from the following claims:

Claim 3.4.4. $\Pr_{\mathsf{Hyb}^0}[\cup_{j\geq 1} E_j] = \Pr_{\mathsf{Hyb}^1}[\cup_{j\geq 1} E_j]$.

Claim 3.4.5. $\mathsf{SD}(\mathsf{Hyb}^1, \mathsf{Hyb}^2) \leq 2\delta$. *Thus,*

$$\Pr_{\mathsf{Hyb}^1}[\cup_{j\geq 1} E_j] \geq \Pr_{\mathsf{Hyb}^2}[\cup_{j\geq 1} E_j] - 2\delta$$

.

Claim 3.4.6. $\mathsf{Hyb}^2 \equiv \mathsf{Hyb}^3$. *Thus,* $\Pr_{\mathsf{Hyb}^2}[\cup_{j\geq 1} E_j] = \Pr_{\mathsf{Hyb}^3}[\cup_{j\geq 1} E_j]$.

Claim 3.4.7. $\Pr_{\mathsf{Hyb}^3}[\cup_{j\geq 1} E_j] \geq 1 - \lambda$.

3.4.2 Proof of Claims 3.4.4 to 3.4.7

We now complete the proof of Lemma 3.4.3 by proving Claims 3.4.4 to 3.4.7.

Claim 3.4.4 (Restated)**.** $\Pr_{\mathsf{Hyb}^0}[\cup_{j\geq 1} E_j] = \Pr_{\mathsf{Hyb}^1}[\cup_{j\geq 1} E_j]$.

Proof. Suppose we sample the hybrid distributions Hyb^0 and Hyb^1 using the same oracle $\mathcal{O}$, same randomness for key generation, and the same randomness for the adversary. Then it is easy to see that for any j, the event E_j holds for Hyb^0 iff it holds for Hyb^1 and so is the event $\cup_{j\geq 1} E_j$. This shows that the probability of $\cup_{j\geq 1} E_j$ happening in both distributions is the same. □

Claim 3.4.5 (Restated)**.** $\mathsf{SD}(\mathsf{Hyb}^1, \mathsf{Hyb}^2) \leq 2\delta$. *Thus,*

$$\Pr_{\mathsf{Hyb}^1}[\cup_{j\geq 1} E_j] \geq \Pr_{\mathsf{Hyb}^2}[\cup_{j\geq 1} E_j] - 2\delta$$

.

Proof. Let B be the event that Adv asks a query in $Q(T_0) \setminus Q(L_M)$, where $Q(T_0)$ denotes the queries in the transcript T_0. It is easy to see that conditioned on B doesn't happen Hyb^1 and Hyb^2 are identically distributed. That is because if we use the same randomness for key generation, oracle and the adversary in the sampling of Hyb^1 and Hyb^2, conditioned on B not happening (in both of them), the value of Hyb^1 and Hyb^2 is equal. In particular it shows that the probability of of B is the same in both distributions. Therefore the statistical distance between Hyb^1 and Hyb^2 is bounded by the probability of B. In the following, we show that $\Pr_{\mathsf{Hyb}^2}[B] \leq 2\delta$. In the the following all the probabilities will be in the experiment for Hyb^2.

Let ϵ, δ and M be as in Step 2 of Adv: $\epsilon = \frac{\delta}{qN}$, and $M = \frac{q}{\epsilon\delta}$. We start by showing: $\Pr[C] \leq \delta$ where the event C is defined as

$$C\colon \exists x \notin Q(L_M) \text{ that is obtained in } \mathbf{D}_M \text{ with prob } \geq \epsilon$$

and $\mathbf{D}_i$ is defined, as in Step 2 of Adv to be the distribution of the transcript of the first signature conditioned on the information in L_i.

Proof of $\Pr[C] \leq \delta$**.** For every possible query x to the random oracle, let q_x denote the probability, taken over both the random oracle and the randomness used by Gen and Adv, that x is queried when generating a key and then signing and verifying α_0. Then $\sum_x q_x \leq q$ **(*)** since this sum is the expected number of queries in this process. Let p_x denote the probability that x is learned at some iteration of Step 2. Then, $q_x \geq \epsilon p_x$ **(**)**. Indeed, if A_i is the event that x is learned at the i'th iteration, then since these events are disjoint $q_x = \Pr[x \text{ is queried}] \geq \sum_{i=1}^{M} \Pr[x \text{ is queried} \mid A_i] \Pr[A_i]$. But by definition of the learning process , $\Pr[x \text{ is queried} \mid A_i] \geq \epsilon$ and hence $q_x \geq \epsilon \sum_{i=1}^{M} \Pr[A_i] = \epsilon p_x$. But the event C only occurs if M distinct queries are learned in Step 2. Hence, if it happens with probability more than δ then the

expected number of queries learned, which is $\sum_x p_x$, is larger than δM. Yet combining **(*)** and **(**)**, we get that $\delta M < \sum_x p_x \leq \sum_x q_x/\epsilon \leq q/\epsilon$, contradicting the fact that $M = q/(\epsilon\delta)$.

Now we will show that $\Pr[B \mid \neg C] \leq \delta$, and it means that $\Pr[B] \geq \Pr[\neg C]\Pr[B \mid \neg C] \geq (1-\delta)^2 > 1 - 2\delta$. Note that Adv makes all its operations in Step 4 based solely on the information in L_M, and the answers chosen for queries $Q(T_0)\backslash Q(L_M)$ does not affect it (because even if queries in $Q(T_0)\backslash Q(L_M)$ are asked by Adv, they will be answered at random). So, it means that the value of Hyb^2 is independent of T_0, conditioned on knowing L_M. Thus, instead of thinking of T_0 being chosen first, then L_M computed and then all queries of Step 4 being performed, we can think of L_M being chosen first, then Adv runs Step 4 based on L_M to sample Hyb^2, and then T_0 is chosen conditioned on L_M and Hyb^2. But because of the independence of T_0 and Hyb^2 conditioned on L_M, the distribution of T_0 conditioned on L_M and Hyb^2 is that conditioned on only L_M which has the distribution $\mathbf{D}_M$. Now assume that L_M makes the event $\neg C$ happen (note that C is defined by L_M.). Since at most qN queries are made in Step 4, and C has not happened, when T_0 is chosen from $\mathbf{D}_M$, the probability that $Q(T_0) \setminus Q(L_M)$ contains one of these queries is at most $\epsilon qN = \delta$. Therefore we get $\Pr[B \mid \neg C] \leq \delta$, and $\Pr[B] < 2\delta$. □

Claim 3.4.6 (Restated)**.** $\mathsf{Hyb}^2 \equiv \mathsf{Hyb}^3$. *Thus,* $\Pr_{\mathsf{Hyb}^2}[\cup_{j\geq 1} E_j] = \Pr_{\mathsf{Hyb}^3}[\cup_{j\geq 1} E_j]$.

Proof. In the sampling of Hyb^3 we can think of L_M being chosen first (although not needed), and then T_0 being chosen conditioned on L_M (i.e., from the distribution $\mathbf{D}_M$), and then Step 4 of the experiment is done while any query in $Q(L_M) \cup Q(T_0)$ is answered according to L_M, T_0, and any other query is answered randomly. (That is we sample L_M and T_0 in the reverse order.) The point is that during the sampling process of Hyb^2 we are also doing exactly the same thing. Again, we sample L_M first. Then $\widetilde{T}_0$ is chosen from the distribution $\mathbf{D}_M$. Then Step 4 is done while any query in

$Q(L_M) \cup Q(\widetilde{T}_0)$ is answered according to $L_M, \widetilde{T}_0$, and all other queries (even the ones in $Q(T_0) \backslash Q(L_M)$) are answered randomly. Therefore Hyb^2 and Hyb^3 have the same distribution. □

Claim 3.4.7 (Restated)**.** $\Pr_{\mathsf{Hyb}^3}[\cup_{j \geq 1} E_j] \geq 1 - \lambda$.

Proof. We will prove that this holds for every fixed oracle and randomness of all parties, as long as the permutation $\alpha_0, \ldots, \alpha_{N-1}$ is chosen at random. For every fixing of the oracle and randomness and $j \in [0..N-1]$, let $U_j = G \cup S_{\beta_j}$ denote the set of queries made by either the key generation algorithm or the signing algorithm for message β_j, and let V_j be the set of queries made by the verification algorithm while verifying this signature. The proof will follow from this fact:

Lemma 3.4.8 (Combinatorial Lemma)**.** *If $U_1, \ldots, U_K, V_1, \ldots, V_K$ are subsets of some universe satisfying $|U_i| + |V_i| \leq q$ and $U_i \cap V_j \nsubseteq V_i$ for every $i \neq j$ then $K \leq \binom{q}{q/2}$.*

The Combinatorial Lemma immediately implies Claim 3.4.7. Indeed, for every i, j with $i \neq j$, define the event $E_{i,j}$ to hold if $U_i \cap V_j \subseteq V_i$. Then, there must be at least $N - \binom{q}{q/2} = N(1-\lambda)$ number of i's (i.e., $1-\lambda$ fraction of them) such that $E_{i,j}$ holds for some j (otherwise we could remove all such i's and obtain a larger than $\binom{q}{q/2}$-sized family contradicting the combinatorial Lemma). But, if we choose a permutation $\alpha_0, \ldots, \alpha_{N-1}$ such that $\alpha_0 = \beta_i$ for such an i then the event $\cup_j E_j$ holds.

Thus, all that is left is to prove is the combinatorial lemma. It essentially follows from Bollobás's Inequality [29], but we repeat the argument here for completeness. Assume for the sake of contradiction that there is a family $U_1, \ldots, U_K, V_1, \ldots, V_K$ satisfying conditions of the lemma with $K > \binom{q}{q/2}$. First, we can remove any elements from U_i that are also in V_i, since it will not hurt any of the conditions. It means that now we have: for every i, j, $U_i \cap V_j = \varnothing$ iff $i = j$. Now, take a random ordering of the universe $W = \bigcup_i (U_i \cup V_i)$, and let A_i be the event that all the

members of U_i occur before the members of V_i in this order. The probability of A_i is $\frac{|U_i|!|V_i|!}{(|U_i|+|V_i|)!} = 1/\binom{|U_i|+|V_i|}{|V_i|} \geq 1/\binom{q}{|V_i|} \geq 1/\binom{q}{q/2}$. Hence if $K > \binom{q}{q/2}$, there is a positive probability that both A_i and A_j hold for some $i \neq j$. But it is not hard to see that in that case, either U_i and V_j are disjoint or U_j and V_i are disjoint, contradicting our hypothesis. □

3.5 A One-Time Signature Scheme

The following theorem shows that Theorem 3.1.1 is tight up to a constant factor in the number of queries.

Theorem 3.5.1. *There is a one-time signature scheme* (Gen, Sign, Ver) *for messages* $\{0,1\}^*$*, using a total of* q *queries to a random oracle that has security* $2^{(0.812-o(1))q}$*, where* $o(1)$ *is a term tending to* 0 *with* q*.*

Proof. The scheme is basically Lamport's Scheme [112] with two changes: **(1)** we use a more efficient anti-chain (family of incomparable sets) than Lamport's scheme (a well-known optimization) and **(2)** we use a secret "salt" value for the hash function to prevent a birthday attack.

The scheme description. Let $c = (3-\sqrt{5})/2$ and k be such that $(1+c)k+4 = q$.

- Generate the keys by choosing k random strings $x_i \in \{0,1\}^{q+i}$ for $0 \leq i \leq k-1$, and an additional random string $z \in \{0,1\}^{2q}$. [10] The secret key consists of these values, and the public key is $\mathcal{O}(x_1), \ldots, \mathcal{O}(x_k), \mathcal{O}(z)$.

- Let $h(\alpha)$ be the first $\log \binom{k}{ck}$ bits of $\mathcal{O}(z, \alpha)$, which we identify with a ck-sized subset of $0, \ldots, k-1$. The signature of α consists of $\{x_i\}_{i \in h(\alpha)}$ and the string

[10] If we choose all of them from $\{0,1\}^q$ the scheme is still as secure as we claimed, but now the analysis is simpler.

z.

- To verify a signature, we first verify that $O(z)$ is equal to its alleged value, then we ask $\mathcal{O}(z, \alpha)$ to know $h(\alpha)$, and then we ask ck more queries to check that the released strings are indeed preimages of the corresponding entries of the public key indexed by $h(\alpha)$.

The number of queries is $q = (1 + c)k + 4$, while, as we will see, the security is at least $\Omega(\binom{k}{ck}) = 2^{(H(c)-o(1))k} = 2^{\frac{H(c)-o(1)}{1+c}q} > 2^{(0.812-o(1))q}$ where $H(\cdot)$ is the Shannon entropy function.

Let T be the total number of oracle queries asked by the adversary and $\alpha \neq \beta$ be (in order) the message for which she asks a signature and the message for which she tries to forge a signature. We assume without loss of generality that $T < 2^{q-1}$, because $2^{q-1} \gg \binom{k}{ck}$. We divide the winning cases for the adversary into three cases:

1. The adversary chooses some $z' \in \{0,1\}^{2q}, z' \neq z$ such that $\mathcal{O}(z) = \mathcal{O}(z')$, alleged to be the real z in the signature of β.

2. The adversary uses the real z in the signature of β and $h(\alpha) = h(\beta)$.

3. The adversary uses the real z in the signature of β and $h(\alpha) \neq h(\beta)$.

We will show that the probability that the adversary wins conditioned on being in case 3 is at most $O(T/\binom{k}{ck})$, and the probability that either case 1 or case 2 happens at all is also at most $O(T/\binom{k}{ck})$. So, the total probability of winning for the adversary will be at most $O(T/\binom{k}{ck})$ as well.

In case 1, even if we reveal z to the adversary in the first place (x_i's are irrelevant), she has the chance of at most $(1+T)/2^q$ to find some $z' \neq z$ such that $\mathcal{O}(z) = \mathcal{O}(z')$. That is because she gets to know at most T oracle query/answer pairs (other than

$\langle z, \mathcal{O}(z)\rangle$), and the probability that she gets $\mathcal{O}(z)$ in one of them is at most $T/2^q$. If she does not see $\mathcal{O}(z)$ as an oracle answer, she needs to guess z' blindly which succeeds with probability at most $1/2^q$.

In the case 2, we reveal all x_i's to the adversary at the beginning, although they are indeed irrelevant to finding a pair $\alpha \neq \beta$ such that $h(\alpha) = h(\beta)$ (because they are of length $< 2q$). Before the adversary gives us α, it asks at most T queries of length $2q$. So, the probability that she gets some $z' \in \{0,1\}^{2q}$ such that $\mathcal{O}(z') = \mathcal{O}(z)$ is at most $T/2^{2q} = o(T/\binom{k}{ck})$. Let assume that this has not happened. So, we can pretend that when we receive α, the value of z is chosen at random different from the members of $\{0,1\}^{2q}$ that are asked from the oracle by Adv. Thus, the probability that any adversary's query so far with length more than $2q$ has the prefix z will be at most $T/(2^q - T) < T/2^{q-1} = O(T/\binom{k}{ck})$. It means that with probability $1 - O(T/\binom{k}{ck})$, so far were no query asked from the oracle which has z as prefix. Assuming this is the case, when we ask the query (z, α) from the oracle, $h(\alpha)$ is chosen uniformly at random from $\{0,1\}^{\log\binom{k}{ck}}$. Hence, if the adversary asks T more oracle queries of the form (z, γ) where $\gamma \neq \alpha$, one of them will give $h(\gamma) = h(\alpha)$ with probability at most $T/\binom{k}{ck}$, and if it does not happen for any of them, a blind guess β by the adversary will give $h(\alpha) = h(\beta)$ with probability $1/\binom{k}{ck}$. So, the probability of getting $\alpha \neq \beta$, $h(\alpha) = h(\beta)$ for the adversary is at most $O(T/\binom{k}{ck})$.

In the case 3, there always is some $i \in h(\beta)\backslash h(\alpha)$. We choose the smallest such i, call it i_0, and change the game slightly by revealing z to Adv from the beginning and revealing all x_j's for $j \neq i_0$ to the her after she gives us β. It only might increases her chance of success (although they are irrelevant because they have different length). For any fixed $i \in 0, \ldots, k-1$, we show that the probability of the adversary to find a preimage for $\mathcal{O}(x_i)$ conditioned on $i = i_0$ is at most $(T+1)/2^{q+i_0} < (T+1)/2^q$ (which is necessary for her to win), and then by the union bound, the probability of

success for the adversary in this case will be at most $k(T+1)/2^q = O(T/\binom{k}{ck})$. The reason is that the adversary can ask at most T oracle queries after we reveal in order to find a preimage for $\mathcal{O}(x_{i_0})$. The probability that for one of the queries x among these T queries she asks we have $\mathcal{O}(x_i) = \mathcal{O}(x)$ is at most $(T)/2^{q+i_0}$, and when it does not happen, the adversary has to guess a preimage for $\mathcal{O}(x_i)$ blindly, which will be correct with probability $1/2^{q+i_0}$.

□

The constant c in the description of the scheme maximizes $\binom{k}{ck}$, conditioned on $q \approx (1+c)k$. The same ideas show that whenever $n \leq dq$ where $d \approx 0.812$ is obtained as above ($d = H(c)/(1+c)$), then there is a one-time signature scheme for messages $\{0,1\}^n$ that makes only q queries and achieved security exponential in the length of its queries.

3.6 Extensions

Now we prove several extensions of Theorem 3.1.1.

Other oracles Using minor changes to the proof of Theorem 3.4.1 we can get a similar lower-bound for signature schemes based on the ideal cipher or a random permutation oracles. This is important as these oracles are also sometimes used to model highly efficient symmetric-crypto primitives, and so it is an interesting question whether such oracles can be used to construct signatures more efficiently.

Theorem 3.6.1. *Let $\mathcal{O}$ be either the ideal cipher oracle. Then, for every one-time signature scheme for messages $\{0,1\}^n$ using a total of $q \leq n/4$ queries to $\mathcal{O}$ there is an adversary making $2^{(4-o(1))q}$ oracle queries that breaks the scheme with probability $1 - o(1)$, where $o(1)$ denotes a term tending to 0 with q. In case of $\mathcal{O}$ being the*

random permutation oracle, only $q \leq n/2$ is needed to get and adversary asking $2^{(2-o(1))q}$ queries, breaking the scheme with probability $1 - o(1)$.

Proof. We explain the proof for the ideal cipher oracle. Extending the proof for the random permutation oracle is straightforward.

We change both the signature scheme and the oracle for the sake of the analysis. We let the new oracle $\mathcal{O}'$ be the same as $\mathcal{O}$ except that $\mathcal{O}'$ does not answer queries of the form (k, x, d) whenever $|x| < 2(q + \log q)$. Instead it answers queries of the type (k, n) where $n < 2(q + \log q)$, to which it returns the long string containing the concatenation of $\mathcal{O}(k, x, \mathsf{F})$ for $x \in \{0,1\}^n$.

We change the signature scheme to get a new scheme $(\mathrm{Gen}', \mathrm{Sign}', \mathrm{Ver}')$ as follows: **(1)** use $\mathcal{O}'$ instead of $\mathcal{O}$ and **(2)** whenever an algorithm makes a query (k, x, d) and obtains an answer y, it will also make the "redundant" query $(k, y, \bar{d})$ (where $\bar{d} = \mathsf{B}$ if $d = \mathsf{F}$ and vice versa). Note that the total number of queries of the new scheme is at most $q' = 2q$.

Lemma 3.6.2. *Given the scheme $(\mathrm{Gen}', \mathrm{Sign}', \mathrm{Ver}')$, there is an adversary Adv making at most $\mathrm{poly}(q')2^{q'}$ queries from $\mathcal{O}'$ that breaks the scheme with probability $1-o(1)$.*

Lemma 3.6.2 implies Theorem 3.6.1 since any such adversary can be implemented using the oracle $\mathcal{O}$ with at most a $q^2 2^{2q}$ factor increase in the number of queries, and the total number of queries will be $\mathrm{poly}(q')2^{q'}q^2 2^{2q} = 2^{(4-o(1))q}$.

Proof. The description of the attack remains basically the same as that of Theorem 3.4.1 set by parameters in Corollary 3.4.2 (i.e. $N = 2^q, \lambda = \delta = \theta(q^{-1/2})$), and we have the same distributions $\mathsf{Hyb}^0, \mathsf{Hyb}^1, \mathsf{Hyb}^2, \mathsf{Hyb}^3$ as before. However, there are some minor changes as follows:

- During Step 2 of the attack, whenever learn a query, we add both the query and its dual to L_i.

- During Step 4 of the attach we might discover an inconsistency between the guesses we made in the sampled transcript $\widetilde{T}_0$ and the answers we receive from the oracle $\mathcal{O}$. That is, we might get the same answer for two different plain texts with the same key. However, as we will see this will only happen with small probability, and we ignore this case safely. .

- The definition of Hyb^2 needs to change a little. Namely, in the experiment for the distribution Hyb^2, during the signing and verification of $\alpha_1, \ldots, \alpha_N$, whenever we make a new non-redundant query (k, x, d), we look at all queries of the form $(k, \cdot, d)$ appearing either in the transcript of the system so far (i.e. $\widetilde{T}_0$, T_1, ...) or in the learned queries of L_M. Then we choose a random answer y from the set of unused answers and use it as the oracle answer for (k, x, d). The next redundant query $(k, y, \bar{d})$ is simply answered by x.

The differences between the proof in this case and the proof of Theorem 3.4.1 are the following:

- We need to include the condition in the event E_j that the queries made in the j'th signing and verification are consistent with (the key generation part of) the transcript $\widetilde{T}_0$ in the sense that they do not specify two queries $(k, x, d), (k, x', d)$, $x \neq x'$ which map to the same answer y. The consistency condition guarantees (by definition) that if E_j occurs, then the verifier will accept the j'th signature.

  The combinatorial condition $V_j \cap (\widetilde{G} \cup \widetilde{S}_0) \subseteq V_0$ still guarantees that the j'th verification does not ask any query for which we have guessed the answer.[11]

  We can still prove that $\Pr_{\mathsf{Hyb}^0}[\exists_j E_j] = \Pr_{\mathsf{Hyb}^1}[\exists_j E_j]$ using basically the same proof as in Claim 3.4.4. We just have to note that as long as E_j happens in both

[11] This also guarantees that there is no inconsistency between the j'th verification and the transcript $\widetilde{T}_0$, but later we will show that the total consistency happens with good probability

experiments, there is no way to distinguish their j'th signing and verification, and the consistency also happens either in both or in none of them. s

- We again show $\mathsf{SD}(\mathsf{Hyb}^1, \mathsf{Hyb}^2) = o(1)$. The reason is that the difference between the distributions Hyb^1 and Hyb^2 is due to some events which happen with probability $o(1)$. That is there are events in the experiments of sampling Hyb^1 and Hyb^2 which happen with probability $o(1)$ and conditioned on they not happening, Hyb^1 and Hyb^2 have the same distribution.

  - Similar to Claim 3.4.5 one of the differences between the distributions $\mathsf{Hyb}^1, \mathsf{Hyb}^2$ might be because of Adv asking a query in $Q(T_0) \setminus Q(L_M)$. Because of the same analysis given in the proof of Claim 3.4.5 the probability that we ask any such query (in both experiments) is at most $2\delta = o(1)$. So, in the following we assume that this case does not happen.

  - In experiment of sampling Hyb^1, when a new non-redundant query (k, x, d) is asked in the $1 \leq i$'th signing or verification, the returned answer y might be equal to a guessed answer for a query (k, x', d) of $\widetilde{T}_0$ (we call this event F_1), but it is never equal to the answer of a query $(k, x'', d) \in Q(T_0) \setminus Q(L_M)$. The situation for Hyb^2 is the reverse: On a new non-redundant query (k, x, d) during the $1 \leq i$'th signing or verification, the answer is never equal to a guessed answer for a query (k, x', d) in $\widetilde{T}_0$, but it might be equal to the answer of a query $(k, x'', d) \in Q(T_0) \setminus Q(L_M)$ (we call this event F_2). Note that $(\mathsf{Hyb}^1 \mid \neg F_1) \equiv (\mathsf{Hyb}^2 \mid \neg F_2)$. As we will see, $\Pr[F_i] = o(1)$ for $i = 1, 2$ which shows that $\mathsf{SD}(\mathsf{Hyb}^1, \mathsf{Hyb}^2) = o(1)$.

    The reason for $\Pr[F_1] = o(1)$ is that whenever we have a new non-redundant query in the $1 \leq i$'th signing or verification, its answer is chosen from a set of size at least $q^2 2^{2q} - q' 2^{q'}$ which might hit a guessed answer for a query in

$\widetilde{T}_0$ with probability at most $q'/(q^2 2^{2q} - q' 2^{q'}) = o(1)$. The same argument holds for $\Pr[F_2] = o(1)$.

- Claim 3.4.6 still holds with the similar proof because of the way we defined Hyb^2 for the case of ideal cipher.
- Claim 3.4.7 is still correct with the same proof. Note that all the signing and verifications are consistent.

□

A similar and simpler proof works for the case of a random permutation oracle. In this case, we again change the oracle by merging small queries into a single query with a huge answer, but we don't have the issue of adding "dual" queries, and therefore the condition $q \leq n/2$ (rather than $q \leq n/4$) is enough to get an adversary who breaks the scheme with probability $1 - o(1)$ by asking $2^{(2-o(1))q}$ queries (rather than $2^{(4-o(1))q}$ queries). □

Implementing Adversary in $\mathbf{BPP^{NP}}$. If the signature scheme is efficient, using an **NP** oracle, our adversary can run in time $\mathrm{poly}(n, 2^q)$, where n is the length of messages to be signed.[12] That is, we prove the following lemma:

Lemma 3.6.3. *If the signature scheme is efficient, the adversary of the proof of Theorem 3.4.1 can be implemented in* $\mathrm{poly}(n, 2^q)$ *time using an oracle to an* **NP***-complete problem.*

[12] In general, the security parameter could be different from the length of the messages n. For example, in Section 3.5, the security parameter was q (so the security was $2^{\Omega(q)}$), and the running time of the algorithms was $\mathrm{poly}(n, q)$. Here, for simplicity, we assume that $\ell = \mathrm{poly}(n)$, and all the algorithms' queries are of length ℓ.

Lemma 3.6.3 can be interpreted as saying that a non-black-box proof of security for a signature scheme more efficient than the lower-bounds provided by Theorem 3.4.1 will necessarily imply a proof that $\mathbf{P} \neq \mathbf{NP}$.

The only place in which the adversary uses its unbounded computational power is in Step 2 where it chooses x_i to be the lexicographically first unlearned string in $\{0,1\}^l$ such that x_i is queried in $\mathbf{D}_i$ with probability at least ϵ, and in Step 3 when it samples a random $\widetilde{T}_0$ from $\mathbf{D}_M$.

We show that:

- Using an **NP** oracle, we can sample from a distribution $\mathbf{D}'_i$ in expected $\text{poly}(n, 2^q)$ time such that $\mathsf{SD}(\mathbf{D}'_i, \mathbf{D}_i) \leq \epsilon$, where ϵ is as defined in Step 2.
- Using the $\mathbf{D}'_i$ sampler, we can implement the adversary in $\text{poly}(n, 2^q)$ time with similar success probability.

We first show how to use a $\mathbf{D}'_i$ sampler to implement the adversary efficiently and then will show how to sample from $\mathbf{D}'_i$ efficiently using an **NP** oracle.

Efficient adversary using a $\mathbf{D}_i$ approximate-sampler. So, here we assume that we can sample efficiently from a distribution $\mathbf{D}'_i$ such that $\mathsf{SD}(\mathbf{D}'_i, \mathbf{D}_i) \leq \epsilon$. In order to choose x_i in the i'th step of the learning phase, we do the following. Let $m = (l + \log M - \log \delta)/\epsilon^2$. We sample m times from $\mathbf{D}'_i$ to get $D_i^1, \ldots, D_i^m$. Then we choose x_i to be the lexicographically first unlearned query (i.e. not in L_{i-1}) which appears in at least 2ϵ fraction of $Q(D_i^j)$'s.

Claim 3.6.4. *With probability at least $1-\delta$ we get the following: For every $x \in \{0,1\}^l$, and every $1 \leq i \leq M$:*

1. *If $\Pr[x \in Q(\mathbf{D}_i)] \geq 3\epsilon$, then x appears in more than 2ϵ fraction of $Q(D_i^j)$'s.*

2. *If* $\Pr[x \in Q(\mathbf{D}_i)] \leq \epsilon$*, then* x *appears in less than* 2ϵ *fraction of* $Q(D_i^j)$*'s.*

If the event above happens, it means that the learning algorithm learns all the 3ϵ-heavy queries in its M rounds with probability at least $1-\delta$ (using the same argument as before). Therefore we get a weaker, yet strong enough, version of Claim 3.4.6 saying that the $\mathsf{SD}(\mathsf{Hyb}^1, \mathsf{Hyb}^2) \leq 3\delta + \delta + \delta = o(\delta)$.

The Claim 3.6.4 follows from the Chernoff bound. The probability that any specific x violates the claim's condition in any of the rounds is at most $e^{-m\epsilon^2} < 2^{-m\epsilon^2} = 2^{-l-\log M+\log\delta}$. By union bound, the probability that the event is not violated at most $M2^l 2^{-l-\log M+\log\delta} = \delta$.

Sampling $\mathbf{D}'_i$ efficiently using an NP oracle. Note that L_i which captures our knowledge of the system after the i'th round of the learning phase can be encoded with $\text{poly}(n, 2^q)$ bits. The number of random bits used by the adversary till the end of the i'th round of the learning phase is also $\text{poly}(n, 2^q)$. For some technical reason which will be clear later, we add the randomness used by the adversary to the description of L_i. Similarly, any (possible) transcript D which $\Pr[\mathbf{D}_i = D] > 0$ can be represented with $\text{poly}(n, q) < \text{poly}(n, 2^q)$ bits. In the following we always assume that such encodings are used to represent L_i and D.

In order to sample from a distribution close to $\mathbf{D}_i$ we use the following Lemma:

Lemma 3.6.5. *There is a function* $f : \{0,1\}^* \times \{0,1\}^* \mapsto \mathbb{N}$ *which is efficiently computable (i.e. time* $\text{poly}(n, 2^q)$*), with the following properties:*

1. $f(L_i, D) = \lfloor c\mathbf{P}[\mathbf{D_i} = D] \rfloor$ *for some constant* c *depending on* L_i*. So we have* $f(L_i, D) = 0$ *if* $\Pr[\mathbf{D_i} = D] = 0$.

2. $f(L_i, D) \geq 10/\epsilon$ *whenever* $\Pr[\mathbf{D} = D] > 0$ *where* ϵ *is as defined in Step 2.*

Before proving the lemma, we see how it is used.

Corollary 3.6.6. *We can sample from a distribution* $\mathbf{D}'_i$ *such that* $\mathsf{SD}(\mathbf{D}_i, \mathbf{D}'_i) \leq \epsilon$ *in time* $\mathrm{poly}(n, 2^q)$ *(where the time* $\mathrm{poly}(n, 2^q)$ *is independent of* i *for* $1 \leq i \leq M$*).*

Proof. Let $W_i = \{(D, j) \mid 1 \leq j \leq f(L_i, D)\}$ be the set of "witnesses" for L_i, where f is the function in Lemma 3.6.5. Lemma 3.6.5 shows that the relation $R = \{(L_i, w) \mid w \in W_i\}$ is an **NP** relation. It is known [14] that for any **NP** relation, there is a witness-sampling algorithm that given any x, samples one of the witnesses of x uniformly in expected $\mathrm{poly}(|x|)$ time. Therefore, we sample a random $w = (D, j)$ such that $w \in W_i$ in expected $\mathrm{poly}(n, 2^q)$-time, and output D. It is easy to that the distribution $\mathbf{D}'_\mathbf{i}$ of our output has statistical distance at most ϵ from the distribution $\mathbf{D_i}$. □

Proof. (Lemma 3.6.5) Recall that $\mathbf{D}_i$ is the distribution of transcripts T_0 conditioned on the information given in L_i. Let the event $E(L_i)$ be the event that during the running of the system (and our attack) adversary's knowledge about the system and its randomness after the i'th round of the learning is what L_i denotes. Similarly, let $E(D)$ be the event that $D = T_0$ is the case in our experiment. Thus, for every transcript D, $\Pr[\mathbf{D}_i = D] = \Pr[E(D) \mid E(L_i)]$. If we could compute $\Pr[E(D) \mid E(L_i)]$, we could somehow use it in the Lemma 3.6.5, but instead of doing that, we will rather compute $\Pr[E(D) \wedge E(L_i)]$ which is proportional to $\Pr[E(D)|E(L_i)]$ up to a constant factor depending on L_i, and will scale it up to some big integer.

Given L_i and D, in order to compute $\Pr[E(D) \wedge E(L_i)]$, we track the whole experiment from the beginning in the following order:

- Key Generation
- Signing α_0

- The attack (which includes the verification of α_0 as its first step) to the end of the i'th round of the Learning.

At any moment that some coin tossing is involved (i.e. in the key generation algorithm, in the attack, or fin an oracle answer), the result is determined by the description of L_i and D. Thus, we can calculate the probability that given values of L_i and D will be the ones in the real running of the experiment[13]. More quantitatively, during the simulation of the experiment, we receive any specific oracle answer with probability at least 2^{-l} whenever it is a possible answer and the probability of getting a specific random tape for the key generation and the adversary is at least $2^{-\operatorname{poly}(n,2^q)}$. Since the total probability of $\Pr[E(D) \wedge E(L_i)]$ is the multiplication of all those probabilities that we get during the simulation of the system, and because the number of oracle queries that we examine is at most $2^{O(q)}$, we get $\Pr[E(D) \wedge E(L_i)] > 2^{-\operatorname{poly}(n,2^q)}$ whenever $\Pr[E(D) \wedge E(L_i)] \neq 0$. Note that ϵ in the attack is $2^{-O(q)}$. Therefore, for a big enough constant $c = \operatorname{poly}(n, 2^q)$, the function $f(L_i, D) = \lfloor c \Pr[E(D) \wedge E(L_i)] \rfloor$ is computable in time $\operatorname{poly}(n, 2^q)$ and we have $f(L_i, D) > 10/\epsilon$ as well. □

3.6.1 Handling Imperfect Completeness

While the typical definition of a signature scheme stipulates that a valid signature (generated by the signing algorithm with the correct key) should be accepted with probability one, it makes sense to consider (especially in the context of negative results) also signatures where the verifier may reject such signatures with small probability, say 1/10. We are able to extend our result to this case as well:

[13]For the case of ideal cipher or random permutation oracles, we need to keep track of the oracle answers so far during the simulation of the experiment, in order to know what is that probability of receiving a specific answer from the oracle at any point.

Theorem 3.6.7. *For every one-time signature scheme for messages* $\{0,1\}^n$*, accepting correct signatures with probability at least* 0.9 *(over the randomness of the verifier), and asking a total* $q \leq \sqrt{n}/20$ *queries to a random oracle, there is* **(1)** *an adversary making* $2^{(1+o(1))q}$ *queries that breaks the scheme with probability at least* 2^{-q} *and* **(2)** *an adversary making* $2^{O(q^2)}$ *oracle queries that breaks the scheme with constant probability.*

The proof of part **(1)** is a straightforward extension of the proof of Theorem 3.4.1 and so we bring here the proof of part **(2)**:

Lemma 3.6.8. *For every one-time signature scheme with imperfect completeness (i.e., verifier can reject valid signatures with probability at most* 1/10 *over its coins) there is an adversary asking* $2^{O(q^2)}$ *queries that finds with probability* $1 - o(1)$ *a message/signature pair which passes the verification with probability at least* 0.7. [14]

Proof. The main difference between the proof of this lemma compared to that of Theorem 3.4.1 is the way we define the sets V_j's. They are not simply the queries that the verifications ask from the oracle. For sake of analysis, for every j, we define the set V_j to be the set computed by the following process: run the j'th verification algorithm on the generated message/signature pair $m =$ times (for m to be defined later), and let V_j be the set of queries that appeared in at least a $1/(20q)$ fraction of these verifications. Hence, we have $|V_j| \leq 20q^2$. Note that the definition of V_j depends on the oracle used to do the verifications. We will treat the sets V_j's in the analysis similar to what we did to them with their previous definition. So, we define the new parameter $r = 20q^2$ to the upper-bound on $|G| + |S_j| + |V_j|$, while q is still an upper-bound for $|G| + |S_j|$. As we will see, the proof will be similar to that of Theorem 3.4.1 and the parameters are set similar to those of Corollary 3.4.2:

[14] The probability 0.7 could be substituted by any constant less than 0.9 with changing the constants in the proof.

$N = 2^r, \lambda = \delta = \frac{\binom{r}{r/2}}{2^r} = \theta(r^{-1/2}) = \theta(1/q), m = 20^3 q^4, \epsilon = \frac{\delta}{mqN}, M = \frac{q}{\epsilon\delta}$. Other than the parameters, the differences compared to the previous attack are:

1. When obtaining the signature σ_0 in Step 1, we run the verification algorithm m times and record in L_0 all the resulting query/answer pairs.

2. In Step 4 we test q^3 times each generated message/signature pair and output the first signature that passes the verification at least a 0.75 fraction of these q^3 times.

We also define the set U_j to be the set of queries that the j'th verification asks from the oracle with probability at least $1/(10q)$ over its own randomness after we fix the random oracle. Hence we have $|U_j| \leq 10q^2$

We say that E_j holds if (as before) $V_j \subseteq (\widetilde{G} \cup \widetilde{S}_0) \cap V_0$. We also say that the event E holds if $U_j \subset V_j$ for every j.

Claim 3.6.9. *If $E_j \wedge E$ holds, then the j'th signature will be accepted by the verifier with probability at least $0.9 - 0.1 = 0.8$ over the randomness of the verifier.*

Proof. The only way this won't happen is that with probability at least $1/10$, the verifier makes a query in the (at most q-sized) set $\widetilde{G} \cup \widetilde{S}_0 \setminus V_0$. But if this happens, then there is a query in that set that is queried first with probability at least $1/(10q)$, yet because E holds that means that it will be contained in $U_j \subset V_j$, contradicting E_j. □

For any specific $1 \leq j < N$, by Chernoff bound, the probability that the fraction of times that we accept the generated signature for α_j is 0.05 far from its real probability of being accepted by the verifier is at most $e^{-0.05^2 q^3}$ and by union bound, the probability that it happens for some j is at most $2^{20q^2} e^{-q^3/400} = o(1)$. Now suppose

$E_j \wedge E$ holds for some $j = j_0$. So by Claim 3.6.1, firstly we will output a pair of message and signature, and secondly this pair is accepted by the verifier with probability at least 0.7.

Claim 3.6.10. *We have* $\Pr[E] \geq 1 - o(1)$.

Proof. By the Chernoff bound, the probability that a particular member of U_j is not in V_j is at most $e^{-(\frac{1}{20q})^2 m} = e^{-20q^2}$. By union bound over the members of U_j, and j we have $\Pr[(\mathbf{2}) \text{ fails for some } j] \leq 10q^2 2^{20q^2} e^{-20q^2} = o(1)$. □

Now that we know E holds almost always, it only remains to show that with high probability E_j happens for some j. This time we define the four hybrid distributions $\mathsf{Hyb}^0, \mathsf{Hyb}^1, \mathsf{Hyb}^2, \mathsf{Hyb}^3$ a bit different. Instead of putting in Hyb^i the query/answer pairs that we received during one verification, we put in Hyb^i all such pairs that we get at some point during the m times that we run the verification.

The proofs of Claims 3.4.4–3.4.7 also work basically in the same way as before:

- Claim 3.4.4 still holds with the same proof.
- Claim 3.4.5 still holds with the same proof because of the new smaller value of ϵ that we used.
- Claim 3.4.6 still holds with the same proof.
- Claim 3.4.7 is still correct with the same proof because the condition $q \leq \sqrt{n}/20$ guarantees that there is enough room to choose $N \leq 2^n$ different messages in the attack.

So, our adversary asks at most $Nmq + M + Nq^3 = \text{poly}(q)2^r = 2^{O(q^2)}$ queries, and with probability $1 - o(1)$ finds a pair of message/signature passing the verification with probability at least 0.7. □

We note that the combination of all the above extensions holds as well (e.g., we can implement in $\mathbf{BPP}^{\mathbf{NP}}$ an adversary that breaks any signature scheme with imperfect completeness that is based on the ideal cipher).

Efficiency of the verifier Because the signing and verification algorithms are run more often than the key generation, lower-bounds on their own efficiency is still meaningful. In Section 3.5 we saw that the signing algorithm can be very efficient while the total number of queries was almost optimal. Here we show that if we want to get an efficient verifier and exponential security at the same time, it makes the total number of queries to be inefficient.

Theorem 3.6.11. *For every one-time signature scheme for messages $\mathcal{M}$ with total q oracle queries where, if the verification asks at most v, $v \leq q/2$ oracle queries and $|\mathcal{M}| \geq \frac{\binom{q}{v}}{\lambda}$ then there is an adversary asking at most $O(\frac{q^2\binom{q}{v}}{\lambda\delta^2})$ queries that breaks the scheme with probability at least $1 - \lambda - \delta$.*

Before going over the proof note that for any $v, k \in \mathbb{N}$, where $3 \leq v \leq \frac{q}{2}$ (i.e. $1 \leq v - 2 \leq k$ where $v + k + 2 = q$) the scheme of Section 3.5 can be simply changed to get a new scheme in which the verifier asks v queries by revealing $v - 2$ sized subsets of x_i's as the signature rather than ck sized ones. A similar proof to that of Theorem 3.5.1 shows that this new scheme has security $\Omega(\binom{k}{v-2}) = \Omega(\binom{q-v-2}{v-2})$. So, if $v = dq$ for constant d, the maximum security S one can get by asking at most $v = q/d$ queries in verification and q queries totally is bounded as $H(\frac{1}{d-1})(1 - 1/d) - o(1) \leq \frac{\log S}{q} \leq H(\frac{1}{d}) + o(1)$ where $H(\cdot)$ is the Shannon's entropy function and $o(1)$ goes to zero with q.

Proof. (Theorem 3.6.11) The proof is almost the same as that of Theorem 3.4.1. The only difference is in Claim 3.4.7 in which we have a restriction that $|V_j| \leq v$, and

we conclude that $K \leq \binom{q}{v}$. The only difference in the proof of Claim 3.4.7 is that now the event A_i has probability at least $\frac{|U_i|!|V_i|!}{(|U_i|+|V_i|)!} = 1/\binom{|U_i|+|V_i|}{|V_i|} \geq 1/\binom{q}{|V_i|} \geq 1/\binom{q}{v}$ because $v \leq q/2$. □

3.7 Lower-Bounds on Black-Box Constructions

In a construction for signature schemes, one might use a *standard primitive* (e.g., one way function) rather than one with ideal security (e.g., random function). These constructions could have different levels of "black-boxness" discussed thoroughly in [143]. What we will call black-box, is called fully black-box in [143]. Here we give a more quantitative definition of such constructions. For simplicity we only define the black-box constructions of signature schemes from hard one-way functions, and the definition of black-box constructions from other primitives are similar. After giving the formal definitions we will prove strong lower-bounds on the efficiency of signature schemes from symmetric primitives when the construction is black-box.

Definition 3.7.1. Let F_ℓ denote the set of all functions $f\colon \{0,1\}^\ell \to \{0,1\}^\ell$ over ℓ bits. We call a family of functions $\{f_\ell \mid \ell \in \mathbb{N}, f_\ell \in F_\ell\}$, *s-hard* (to invert), if for any probabilistic algorithm A running in time at most $s(\ell)$, we have $\Pr_{x \xleftarrow{R} \{0,1\}^\ell}[A(f(x)) \in f^{-1}(f(x))] \leq \frac{1}{s(\ell)}$ where the probability is over the choice of x and the coin tosses of A.

By S-hard functions, for a *set* of functions S, we mean all those which are s-hard for some $s \in S$. (Think of S as the set of all the functions which are super-polynomial, quasi-polynomial, exponential, etc.) We will keep the notation that the capital leter denotes a set of functions.

For simplicity we use n, the length of the messages to be signed, as the security parameter of the signature scheme (i.e, the efficient schemes will run in time poly(n)

and for larger values of n the scheme becomes more secure).

Definition 3.7.2. A *black-box* construction of one-time signature schemes for n-bit messages from S-hard one-way functions, with security parameter contraction $\ell(n)$ is made of the following two families of reductions for all $n \in \mathbb{N}$ (we don't write the index n for the reductions):

- The implementation reduction $I = (\mathrm{Gen}, \mathrm{Sign}, \mathrm{Ver})$ has three components which are algorithms running in time $\mathrm{poly}(n)$ (Gen is probabilistic) and $I^f = (\mathrm{Gen}^f, \mathrm{Sign}^f, \mathrm{Ver}^f)$ satisfies in Definition 3.3.1 by setting $\mathcal{O} = f$ for any $f \in F_{\ell(n)}$.

- We call A a *I^f-breaker* if A is a (not necessarily efficient) adversary who breaks the security of I^f with non-negligible probability by playing in the game defined in Definition 3.3.2. The security reduction R is an algorithm running in time $t(n)$ where: **(1)**: For any $f \in F_{\ell(n)}$ and any I^f-breaker A, $\Pr_{x \xleftarrow{R} \{0,1\}^{\ell(n)}}[R^{A,f}(f(x)) \in f^{-1}(f(x))] \geq \frac{1}{w(n)}$ where the probability is over the choice of x and the coin tosses of R and A, **(2)**: $t(n)p(n) < s(\ell(n))$ for any $p(n) = \mathrm{poly}(n)$, any $s(\cdot) \in S$ and and large enough n, and **(3)**: $w(n) < s(\ell(n))$ for any $s \in S$ and large enough n.

The security parameter contraction factor $\ell(n)$ in Definition 3.7.2 measures how small the length of the function used in the reduction is (i.e., the security parameter of the primitive used) compared to n (i.e., the security parameter of the signature scheme). The term "security parameter expansion" is used in [82] for the inverse of the contraction parameter here.

Note that having such a reduction, the existence of any efficiently computable family of functions $f\colon \{0,1\}^\ell \to \{0,1\}^\ell$ which is s-hard to invert for some $s \in S$ implies the existence of (efficient) one-time signature schemes which are secure against

polynomial-time adversaries. That is because **(1)**: We get an efficient implementation of the scheme by efficiently implementing f for I^f, and **(2)**: If A is a I^f-breaker running in time $\text{poly}(n)$, the reduction R combined with its subroutine A breaks the s-hardness of f which is not possible.

Now we prove a strong lower-bound on the efficiency of signature schemes relying on the efficiency of strong one-way functions. Then we will show how it generalizes to any symmetric primitive and also functions with many hard-core bits.

Theorem 3.7.3. *Let* $\mathbf{E}$ *denote the set of functions* $\mathbf{E} = \{f(\ell) \mid f = 2^{\Omega(\ell)}\}$. *Any black-box construction of one-time signature schemes for* n*-big messages from* $\mathbf{E}$*-hard one-way functions with security parameter contraction* $\ell(n)$ *needs to ask* $\min(\Omega(\ell(n)), n)$ *queries from the one-way function.*

Before going over the proof we make two observations. First, if construction uses $\mathbf{E}$-hard functions, it means that we should have $t(n) = 2^{o(\ell(n))}$ and $w(n) = 2^{-o(\ell(n)}$ in the security reduction. Another point is that the existence of such a reduction regardless of how many queries it asks, makes $\ell(n)$ to be $\omega(\log n)$ for otherwise the condition $t(n)\,\text{poly}(n) < s(\ell(n))$ in Definition 3.7.2 will be violated. Therefore without loss of generality, we assume that $q \geq \log n$, because otherwise we can ask $\log n$ redundant queries in the key generation algorithm without changing the condition $q \leq \min(\Omega(\ell(n)), n)$.

Proof. For sake of contradiction suppose that there is a black-box construction of signature schemes (I, R) where I asks $q \leq n$ queries from the one-way function and $\log n \leq q = o(\ell(n))$.

The proof will go in two steps. We will first show that any such construction results in a (computationally unbounded) adversary asking $2^{o(\ell)}$ queries from a a random function $f \xleftarrow{R} F_\ell$ and inverting it on a random point with probability at least

$2^{-o(\ell)}$ (where this probability is also over the choice of f). Then we will show that it is not possible to have such an adversary, namely any adversary asking $2^{\ell/3}$ queries has chance of at most $2^{-\ell/3}$ for doing so.

Step 1. Let A be the adversary of Corollary 3.4.2 for the implementation of the signature scheme I (note $q \leq n$) asking at most $2^{(1+o(1))o(\ell(n))}$ queries from the function f (note $q \leq o(\ell(n))$ and breaking I^f with probability at least $1-o(1)$ when f is chosen at random $f \xleftarrow{R} F_{\ell(n)}$ where $o(1)$ goes to zero with q. For large enough $\ell(n)$, n becomes large enough too, and so does q (because $q \geq \log n$). Therefore A asks at most $2^{o(\ell(n))}$ queries from f and breaks the scheme with probability at least $3/4$ when $f \xleftarrow{R} F_{\ell(n)}$ for large enough $\ell(n)$. By an average argument, with probability at least $1/2$ over the choice of f, A breaks I^f with probability at least $1/2$ over its own randomness. We call such f's the good ones. Whenever f is good, $R^{A^f,f}$ inverts f on a random point with probability at least $2^{-o(\ell(n))}$, and because f is good with probability at least $1/2$, $R^{A,f}$ inverts f on a random point with probability at least $2^{-o(\ell(n))}$ for a randomly chosen $f \xleftarrow{R} F_{\ell(n)}$ where the probability is over the choice of f, the choice of the image to be inverted, and the randomness of A. By merging the code of R with A, we get an adversary $B = R^A$ who asks at most $2^{o(\ell(n))}2^{o(\ell(n))} = 2^{o(\ell(n))}$ queries from $f \xleftarrow{R} F_{\ell(n)}$ and inverts it on a random point (i.e., $y = f(x)$ for $x \xleftarrow{R} \{0,1\}^{\ell(n)}$) with probability at least $2^{-o(\ell(n))}$.

Step 2. Suppose B is an adversary asking $2^{\ell/3}$ queries from a random function $f \xleftarrow{R} F_\ell$ trying to find a preimage for $f(x)$ where $x \xleftarrow{R} \{0,1\}^\ell$. We can pretend that the value of f at each point is determined at random whenever it is asked for the first time. So, at first x is chosen, $f(x)$ is chosen, and it is given to B. At first B does not have any information about x, so the probability that B asks x in any of its $2^{\ell/3}$ queries is at most $2^{-2\ell/3}$. Assuming it does not ask x, the probability that B receives

the answer $f(y) = f(x)$ by asking any $y \neq x$ is at most $2^{-2\ell/3}$. Assuming that none of the mentioned events happens, if it outputs y differen from all queries it has asked from f, $f(y) = f(x)$ happens with probability $2^{-\ell}$. So its chance of winning is at most $2^{-2\ell/3} + 2^{-2\ell/3} + 2^{-\ell} < 2^{-\ell/3}$ (for $\ell \geq 4$).

□

As it is clear from the theorem, our lower-bound becomes stronger for larger values of $\ell(n)$ which is also the case in the similar (unconditional) lower-bound results [82, 156].

In order to extend the lower-bound to other symmetric primitives (and functions with many hard-core bits) we can follow the same steps of the proof of Theorem 3.7.3 using the following lemma.

Lemma 3.7.4. *Let P be a symmetric primitive (i.e, one-way function, one-way permutation, collision resistent hash function, pseudorandom generator, pseudorandom function, message authentication code, or block cipher) , or the primitive of functions $f\colon \{0,1\}^{\ell} \to \{0,1\}^{\ell}$ with $\ell/2$ hard-core bits. Then, there is an implementation for P for security parameter ℓ with access to either, random oracle, random permutation oracle, or ideal cipher oracle which asks only a constant number of queries of length $\theta(\ell)$ from the oracle, and any (computationally unbounded) adversary* Adv *who asks at most $2^{o(\ell)}$ queries from the oracle has chance of at most $2^{-\Omega(\ell)}$ of breaking it (over the randomness of* Adv *and the oracle used).*

Proof. We will describe the natural implementations and will show the proof of security only for the case that P is the primitive of functions with $\ell/2$ hard-core bits. The security proofs for other implementations are also easy to get (in fact, we already gave the proof for the case of one-way function in the proof of Theorem 3.7.3).

- **One-way function** using random oracle: To define the value of the function f on input $x \in \{0,1\}^\ell$, we simply use the oracle's answer: $f(x) = \mathcal{O}(x)$.

- **One-way permutation** using random permutation oracle: To define the value of the permutation p on input $x \in \{0,1\}^\ell$, we simply use the oracle's answer: $p(x) = \mathcal{O}(x)$.

- **Collision resistent hash function** using random oracle: The value of the hash function h on input $x \in \{0,1\}^\ell$ is made by using the first $\ell/2$ bits of the oracle's answer: $h(x) = b_1 \dots b_{\ell/2}$ where $\mathcal{O}(x) = b_1 \dots b_\ell$.

- **Pseudorandom generator** using random oracle: The stretched output of the generator g on input $x \in \{0,1\}^\ell$ is the output of the oracle on the padded query: $g(x) = \mathcal{O}(x|0^\ell)$.

- **Pseudorandom function** using random oracle: Using the key $k \in \{0,1\}^\ell$ on input $x \in \{0,1\}^\ell$, the output of the function will be the first ℓ bits of the oracle's answer on the query made by attaching k and x: $f_k(x) = b_1 \dots b_\ell$ where $\mathcal{O}(k|x) = b_1 \dots b_{2\ell}$.

- **Message Authentication Code** using random oracle: Using the key $k \in \{0,1\}^\ell$, the authentication code of the message $x \in \{0,1\}^\ell$ is defined similar to that of pseudorandom function. The verification is clear.

- **Block cipher** using ideal cipher oracle: Using the key $k \in \{0,1\}^\ell$ and the input $x \in \{0,1\}^\ell$, and the direction d we simply use the oracle's answer $\mathcal{O}(k, x, d)$ as our cipher.

- **Function with $\ell/2$ hard-core bits** using random oracle: The value of the function f on input $x = x_1 \dots x_\ell$ uses the oracle's answer: $f(x) = \mathcal{O}(x)$ and the hard-core bits for x will be the first $\ell/2$ bits of it: $HC(x) = x_1 \dots x_{\ell/2}$.

Now we prove the claim for the last primitive (i.e., functions with $\ell/2$ hard-core bits). Suppose the adversary A asks at most $2^{\ell/4}$ queries from the function f. Again, we assume that f chooses its answers randomly whenever asked for the first time. In order to break the hard-core property of the function f, the adversary A needs to distinguish between two experiments. In the first one she is given $(f(x), U_{\ell/2})$ as in put, and in the second one she is given $(f(x), HC(x))$, and in both of the experiments $f \xleftarrow{R} F_\ell$ and $x \xleftarrow{R} \{0,1\}^\ell$ are chosen at random. Note that as long as A does not ask x from the oracle, the two experiments are the same. At the beginning A does not knows the second half of the bits of x. So the probability that she asks x from the oracle in one of her $2^{\ell/4}$ queries is at most $2^{\ell/4}2^{-\ell/2} = 2^{-\ell/4}$. Hence, if the probability that she outputs 1 in the experiment i is p_i (for $1 \leq i \leq 2$), we have $|p_1 - p_2| \leq 2^{-\ell/4}$.

□

So by using Lemma 3.7.4 and following the steps of the proof of Theorem 3.7.3 we get the following theorem:

Theorem 3.7.5. *Let* $\mathbf{E}$ *denote the set of functions* $\mathbf{E} = \{f(\ell) \mid f = 2^{\Omega(\ell)}\}$*, and* P *be either a symmetric primitive or the primitive of functions with* $\ell/2$ *hard-core bits. Any black-box construction of one-time signature schemes for* n*-bit messages from an* $\mathbf{E}$*-hard primitive* P *with security parameter contraction* $\ell(n)$ *needs to ask* $\min(\Omega(\ell(n)), n/4)$ *queries from the primitive* P*.*

3.8 Conclusions and Open Questions

We believe that lower-bounds of this form— the efficiency of constructing various schemes using black box idealized primitives— can give us important information on the efficiency and optimality of various constructions. In particular, three natural questions related to this work are:

- Can one pinpoint more precisely the optimal number of queries in the construction of one-time signature schemes based on random oracles? In particular, perhaps our lower-bound can be improved to show that the variant of Lamport's scheme given in Section 3.5 is optimal up to lower order terms.

- What is the threshold d that whenever $n \leq dq$, we can get signature schemes for messages $\{0,1\}^n$ using q oracle queries, and arbitrary large security? Again, it seems that the variant of Lamport's scheme given in Section 3.5 (working for $\log \binom{k}{ck}$ bit messages without hashing) gives this threshold (i.e., $d \approx 0.812$).

- Can we obtain a $2^{O(q)}$-query attack succeeding with high probability against signature schemes with imperfect completeness?

- Are there stronger bounds for *general* (not one-time) signatures? A plausible conjecture is that obtaining a T-time signature with black-box security S requires $\Omega(\log T \log S)$ queries.

Part II

NP-Hard Cryptography

Chapter 4

Sampling with Size and Applications to NP-Hard Cryptography

4.1 Introduction

The ability to sample from efficiently decidable sets (i.e., membership in such a set can be decided efficiently, but sampling from the set might be hard) is an extremely powerful computation resource, to the point that having such ability for *any* decidable set implies $\mathbf{P} = \mathbf{NP}$. In this work we study less powerful samplers, which only agree to sample from more carefully chosen sets. We show that while these samplers can be used to break certain cryptographic primitives, they seem not to be strong enough to decide arbitrary **NP** languages. We then use this fact to give negative evidence on the possibility of basing such primitives on **NP** hardness.

Consider the sampler that gets a circuit C over $\{0,1\}^n$ as input, and outputs two random values x and x' in $\{0,1\}^n$ conditioned that $C(x) = C(x')$. Such a sampler

is known as a "collision finder", and breaks the security of any family of collision-resistant hash functions [150].[1] We consider the following generalization of the above sampler: the sampler Sam_d, where $d \in \mathbb{N}$, gets up to d recursive calls, each of the form $(C_1, \ldots, C_i, x)$, where $i \leq d$, each of the C_j's is a circuit over $\{0,1\}^n$ and $x \in \{0,1\}^n$. Sam_d answers depth 1 calls, $(C_1, \cdot)$, with a random element in $\{0,1\}^n$. For depth $i > 1$ calls, Sam_d first checks that it was previously queried with $(C_1, \ldots, C_{i-1}, \cdot)$ and answered with x (otherwise, it aborts). If the check passes, then Sam_d answers with a random element in $C_1^{-1}(C_1(x)) \cap \ldots \cap C_i^{-1}(C_i(x))$. (Note that Sam_2, is equivalent to the "collision finder" we described above). Such a sampler is very powerful, as it can be used for breaking the binding of any d-round statistically hiding commitments [86, 156].

Commitment schemes are the digital analogue of a sealed envelope. In such a scheme, a sender and a receiver run an interactive protocol where a sender commits to a bit b. In case the commitment is statistically hiding, then the protocol guarantees that from the receiver's point of view there exists roughly equal chance that the sender has committed to $b = 0$ or $b = 1$ (hence the bit b is hidden from the receiver information-theoretically). In addition, the scheme guarantees that a computationally-bounded sender can only find one way to decommit. Statistically hiding commitments are widely used throughout all of cryptography, with applications including, but not limited to, constructions of zero-knowledge protocols [12, 33, 66, 87, 125], authentication schemes [44], and other cryptographic protocols (e.g., coin-tossing [114]). Hence, it is highly important to study the minimal assumptions required for building them. Since Sam_d breaks any d-round statistically hiding commitments, it is very informative to learn what hardness assumptions Sam_d does

[1] A family of collision resistant hash functions is a family of, efficient, compressing functions with the following security guarantee: given a random function h in the family, it is hard to find $x \neq x'$ satisfying $h(x) = h(x')$.

not break (in particular, we have little hope to base d-round statistically hiding commitments on such assumptions). The following theorem shows that for a constant d, Sam_d is not "too powerful".

We write $L \in \mathbf{BPP}^{\mathcal{O}[k]}$ to mean that L can be decided by $A^{\mathcal{O}}$, where A is a *k-adaptive* (randomized) oracle algorithm using an oracle $\mathcal{O}$: A makes k adaptive rounds of queries to its oracle; each round may consist of many queries, but all of the queries in one round can be computed without looking at the oracle responses to any of the other queries in the same or later rounds. We say A is non-adaptive if $k = 1$.

Theorem 4.1.1 (Main theorem, informal)**.** *For any $d = O(1)$ and any efficient oracle algorithm A, there exists an interactive protocol* AM–Sam *with the following guarantee: either the output of the efficient verifier is statistically close to the output of A^{Sam_d}, or (if the prover cheats) the verifier aborts with high probability. Furthermore, the round complexity of* AM–Sam *is the same as the adaptivity of the oracle queries of A, and the honest prover strategy has complexity $\mathbf{BPP}^{\mathbf{NP}}$ (while the protocol remains sound against unbounded cheating provers).*

We apply this theorem to understand what languages can be efficiently decided by randomized oracle algorithms with oracle access to $\mathsf{Sam}_{O(1)}$, where the strength of the implication is a result of the adaptivity of the calls to $\mathsf{Sam}_{O(1)}$ made by the algorithm. Theorem 4.1.1 yields a k-round protocol for any language $L \in \mathbf{BPP}^{\mathsf{Sam}_{O(1)}[k]}$. Since $\mathbf{BPP}^{\mathsf{Sam}_{O(1)}[k]}$ is closed under complement, the above implies the following corollary.

Corollary 4.1.2 (Limits of languages decidable using oracle access to $\mathsf{Sam}_{O(1)}$)**.** *It holds that $\mathbf{BPP}^{\mathsf{Sam}_{O(1)}[k]} \subseteq \mathbf{AM}[k] \cap \mathbf{coAM}[k]$. In particular, every $L \in \mathbf{BPP}^{\mathsf{Sam}_{O(1)}[k]}$ has a k-round interactive proof where the honest prover has complexity $\mathbf{BPP}^{\mathbf{NP}}$. Furthermore, if L is $\mathbf{NP}$-complete, then the following consequences hold.*

- $k = \text{poly}(n)$*:* **co-NP** *has a public-coin* $O(k)$*-round interactive proof with honest prover complexity* $\mathbf{BPP}^{\mathbf{NP}}$*.*
- $k = \text{polylog}(n)$*: the quasipolynomial hierarchy collapses to its third level (by [136]).*
- $k = O(1)$*:* $\mathbf{PH} = \Sigma_2$ *(by [30]).*

Since the polynomial hierarchy is widely conjectured not to collapse, it follows that **NP**-complete languages are unlikely to be in $\mathbf{BPP}^{\mathsf{Sam}_{O(1)}[k=O(1)]}$. For $k = \text{polylog}(n)$, the collapse is less understood, but it is still reasonable to conjecture that such a collapse does not occur. For $k = o(n)$ the consequence may not be implausible, but would nevertheless lead to surprising progress on the long-standing open question of reducing the round complexity of interactive proofs for **co-NP** [115]. Finally for $k = \text{poly}(n)$, as pointed out to us by Holenstein [96], it would answer a long-standing open question of Babai et al. [9] about reducing the complexity of the prover in interactive proofs for **co-NP** from $\mathbf{BPP}^{\#\mathbf{P}}$ to $\mathbf{BPP}^{\mathbf{NP}}$ (in fact this question is even open for multi-prover interactive proofs). Thus, depending on the adaptivity k, Corollary 4.1.2 gives an indication of either the implausibility or the difficulty of proving that **NP**-complete languages can be decided using the help of $\mathsf{Sam}_{O(1)}$.

4.1.1 Application to Basing Cryptography on NP-Hardness

Since $\mathsf{Sam}_{O(1)}$ breaks the security of d-round statically hiding commitments, it also breaks the wide variety of cryptographic primitives that yield such commitments via constant-adaptive black-box reductions. This list includes: collection of claw-free permutations with an efficiently-recognizable index set [66], collision-resistant hash functions [46, 124], (singly) homomorphic encryption [100], constant-round protocols for oblivious-transfer and private information retrieval schemes where the security of

one of the parties holds information theoretically [86], the average-case hardness of **SZK** [130], constant-round statistically *binding* commitments secure against selective opening attacks [157], and constant-round inaccessible entropy generators [88]. The following corollary states that if any of the above primitives can be based on **NP**-hardness via a black-box reduction R, then $R^{\mathsf{Sam}_{O(1)}}$ decides SAT.

Corollary 4.1.3 (Immediate by Corollary 4.1.2)**.** *Let P be a cryptographic primitive whose security can be broken by $\mathsf{Sam}_{O(1)}$. Let R be a k-adaptive reduction that bases the existence of P on $\mathbf{NP}$-hardness. Then* $\mathrm{SAT} \in \mathbf{AM}[k] \cap \mathbf{coAM}[k]$*, where the honest provers that realize this containment are in $\mathbf{BPP}^{\mathbf{NP}}$. The various consequences for different k given in Corollary 4.1.2 also hold.*

We remark that previous results studying the analogous question of basing (general) one-way functions on **NP**-hardness were restricted to non-adaptive reductions [5, 27, 54]. Other works do consider adaptive reductions, but with respect to more structured primitives [5, 31, 64]. See Section 4.1.3 for the description of previous works.

4.1.2 Main Tool—A New Sampling Protocol

Our main tool for proving Theorem 4.1.1, which is also our main technical contribution, is a new constant-round public-coin sampling protocol that we believe to be of independent interest. A distribution D is called *efficiently samplable* if it is the output distribution of an efficient function $f : \{0,1\}^n \mapsto \{0,1\}^*$ (i.e., $D = f(U_n)$). A distribution is efficiently samplable *with post-selection* if $D = f(U_{\mathcal{S}})$ where $U_{\mathcal{S}}$ is the uniform distribution over an efficiently decidable set $\mathcal{S} \subseteq \{0,1\}^n$. Such distributions have also been studied in the context of randomized algorithms [91]. We emphasize that although $\mathcal{S}$ is efficiently decidable, it is not necessarily possible to efficiently

sample uniform elements of $\mathcal{S}$.

Our "Sample With Size" protocol takes $f, \mathcal{S}$, and a good approximation of $|\mathcal{S}|$ as input, and enables an efficient verifier to sample a uniform $y \in f(\mathcal{S})$, along with a good approximation of the value $|f^{-1}(y) \cap \mathcal{S}|$.

Lemma 4.1.4. *(Sampling With Size protocol, informal) There exists a constant-round public-coin protocol* SampWithSize*, where the parties get as a common input an efficiently decidable set $\mathcal{S} \subseteq \{0,1\}^n$, an efficiently computable function $f : \mathcal{S} \mapsto \{0,1\}^*$ and a good approximation (i.e., within $(1 \pm \frac{1}{\mathrm{poly}(n)})$ factor) of $|\mathcal{S}|$, and has the following guarantees:*

Either $\mathsf{V}_{\mathsf{SWS}}$ outputs a pair (x, s_x) such that 1. x is distributed $(1/\mathrm{poly}(n))$-statistically close to the uniform distribution over $\mathcal{S}$, and 2. s_x is a good approximation for $|f^{-1}(f(x)) \cap \mathcal{S}|$, or (if the prover cheats) the verifier aborts with high probability. Furthermore, the honest prover has complexity $\mathbf{BPP}^{\mathbf{NP}}$, while the cheating prover may be unbounded.

4.1.3 Related Work

NP-Hardness and Cryptography

Brassard [31] showed that if there exists a deterministic black-box reduction from **NP**-hardness to inverting a one-way permutation, then **NP**= **co-NP**. Bogdanov and Trevisan [27], building on earlier work of Feigenbaum and Fortnow [54], showed that if there exists a *non-adaptive* randomized black-box reduction from **NP**-hardness to inverting a one-way function (or more generally, to a hard on the average problem in **NP**), then $\mathbf{NP} \subseteq \mathbf{coAM}/\mathrm{poly}$, which is considered implausible since the polynomial hierarchy would collapse to the third level [159]. Akavia et al. [5] improved this result for the case of reductions to inverting one-way functions, to show that the

same hypothesis implies the uniform conclusion $\mathbf{NP} \subseteq \mathbf{coAM}$, which implies that the polynomial hierarchy collapses to the second level [30].

Goldreich and Goldwasser [64] showed that adaptive reductions basing public-key encryption schemes with the special property that the set of invalid ciphertexts is verifiable in **AM**, on **NP**-hardness would also imply that $\mathbf{NP} \subseteq \mathbf{coAM}$. Finally, Pass [135] takes a different route and showed that if a *specific type* of *witness-hiding* protocol exists, then an arbitrarily adaptive reduction from **NP**-hardness to the existence of one-way functions implies that $\mathbf{co\text{-}NP} \subseteq \mathbf{AM} \cap \mathbf{coAM}$. As recently pointed out by Haitner et al. [89], however, it is unlikely that known witness-hiding protocols are of the type required by [135].

We remark that while most cryptographic reductions we know of are non-adaptive, there are a few notable exceptions. In particular, security reductions for building interactive protocols [125], pseudorandom number generators [90, 92, 95], and certain lattice-based cryptosystems [4, 121]. One may hope in particular that lattice problems might someday be used to prove that $\mathbf{P} \neq \mathbf{NP}$ implies one-way functions or collision-resistant hash functions, since they already exhibit a worst-case to average-case hardness reduction.[2] Previous work such as [5, 27] do not rule out the possibility that any of these (or some other adaptive) techniques may succeed.

The Oracle Sam

Simon [150] studied collision finders that break collision-resistant hash functions. In our language of Sam, a collision finder is the same as Sam_2, i.e., Sam where queries of at most depth 2 are allowed. Simon [150] considered the sampler Sam_2^{π} — a generalization of Sam_2 that gets circuits with π-gates, where π is a random permutation

[2] In particular, the adaptivity of the lattice-based schemes seems essential for giving the best known approximation-ratio required in the worst-case hard lattice problem. Unfortunately, even in the best known reductions the starting worst-case hard problem in the $\mathbf{NP} \cap \mathbf{co\text{-}NP}$.

oracle. He showed that while Sam_2^π breaks any collision-resistant hash functions relative to random permutation π (i.e., the hash function is allowed to use π-gates), it cannot invert π. Continuing this line of research, Haitner et al. [86] showed that Sam_d^π breaks all d-round statistically hiding commitments, even those implemented using π, but Sam_d^π does not help to invert π if $d = o(n/\log n)$. As a consequence, the above results rule out the possibility of basing $o(n/\log n)$-round statistically hiding commitments on the existence of one-way functions/permutations, using *fully-black-box* reductions — a reduction from (the security of) a primitive to one-way function is fully-black-box, if the proof of security is black-box (in the sense of Corollary 4.1.3), and *in addition* the construction uses the one-way function as a black-box (i.e., as an oracle). Note that these results are incomparable to the result stated in Corollary 4.1.3. On one hand, they rule out all fully-black-box reductions *unconditionally* without restrictions on adaptivity, and the reductions they consider are starting from one-way functions rather than **NP**-hardness (and thus "harder" to refute). On the other hand, their results do not apply to constructions that may use the code of the one-way function (or, in our case, the structure of the **NP**-complete language). In contrast, Corollary 4.1.3 also applies to reductions where the construction is non-black-box, which permits, for example, the construction to exploit the fact that YES instances of **NP** languages have efficiently verifiable witnesses. In other words, Corollary 4.1.3 only requires that the *security analysis* be black-box. We refer the reader to Reingold et al. [143], which, although not focused on our case where the construction is non-black-box but the security analysis is black-box, is useful for understanding the distinctions between various notions of reductions.

Sam and zero-knowledge. In recent work, Gordon et al. [79] observe that our main result is useful in the context of understanding zero-knowledge proofs. In particular,

they prove using Theorem 4.1.1 that if a language L has a constant-round black-box computational zero-knowledge proof based on one-way permutations with a k-adaptive simulator, then $L \in \mathbf{AM}[k] \cap \mathbf{coAM}[k]$. Their result suggests that reducing the round complexity of known constructions of zero-knowledge proofs based on one-way permutations for **NP** (e.g., [22, 71]) (all of which have super-constant round complexity) to a constant number of rounds is implausible (if the simulator must be $O(1)$-adaptive) or at least difficult to prove (regardless of the simulator's adaptivity).

Efficiently Samplable Distributions with Post-Selection

Estimating statistics. Estimating statistical properties *efficiently samplable* distributions has long been studied in theoretical computer science [2, 56, 68, 75, 129, 146]. Typically, estimating interesting parameters of samplable distributions (and therefore also of samplable distributions with post-selection) such as entropy or statistical difference is hard (e.g., **SZK**-hard). Nevertheless, for samplable distributions it was known that an efficient verifier can estimate various parameters in constant rounds with the help of an all-powerful prover.

Bounding set-size protocols. The constant-round public-coin lower-bound protocol of Goldwasser and Sipser [75] can be used to lower-bound the size of efficiently decidable sets. Namely, on input an efficiently decidable set $\mathcal{S}$ and a value s, the prover makes the verifier accept iff $|\mathcal{S}| \geq s$. Fortnow [56] (see also Aiello and Håstad [2]) gives a constant-round protocol that upper-bounds the sizes of efficiently decidable sets $\mathcal{S}$ where *in addition* the verifier has a uniform element of $\mathcal{S}$ that is *unknown* to the prover.

These protocols are related to our protocol SampWithSize. For example, one can estimate with respect to $D = f(U_n)$ the integer $s_y = |f^{-1}(y)|$ for a random $y \xleftarrow{R} D$ by

lower-bounding and upper-bounding the set $|f^{-1}(y)|$. In particular, the upper-bound [2, 56] can be applied in this case, since the verifier can sample $x \stackrel{R}{\leftarrow} U_n$, compute $y = f(x)$ and ask the prover for an upper-bound on the size of the set $f^{-1}(y)$ without revealing x. This is one way to prove SampWithSize for the special case $\mathcal{S} = \{0,1\}^n$.

We cannot necessarily apply, however, the upper-bounds of [2, 56] to do the same thing with *post-selected* distributions $D = f(U_{\mathcal{S}})$; even though $f^{-1}(y)$ is efficiently decidable, it may not be possible to efficiently generate $y \stackrel{R}{\leftarrow} f(U_{\mathcal{S}})$ and $x \in f^{-1}(y)$ such that x is hidden from the prover. As we discuss in Section 4.2.2, handling post-selected distributions is necessary to obtain Theorem 4.1.1. Although one-sided lower-bound estimates can be obtained from the lower-bound protocol of [75], it is unknown how to get two-sided estimates using the upper-bound protocol of [2, 56], where the difficulty is to obtain secret samples from $U_{\mathcal{S}}$. In contrast, SampWithSize *does* guarantee a two-sided bound for $|f^{-1}(f(x)) \cap \mathcal{S}|$ for a random x in $U_{\mathcal{S}}$.

Sampling. Using an all-powerful prover to help sample is an old question in computer science, dating at least to the works of Valiant and Vazirani [154] and Impagliazzo and Luby [97]. In building SampWithSize, we use a sampling protocol from Goldreich et al. [74]. This constant-round public-coin protocol takes as input an efficiently decidable set $\mathcal{S}$ and a good approximation of $|\mathcal{S}|$, and outputs a nearly-uniform element of $\mathcal{S}$. Our protocol SampWithSize uses their sampling protocol and extends it by also giving set size information about the sample that is generated.

Another protocol that seems related to SampWithSize is the random selection protocol of Goldreich et al. [69]. Their protocol accomplishes a goal similar to the protocol of [74], allowing a verifier to select a random element of a set. Their protocol, however, cannot be applied in our context as it requires super-constant round complexity. Other related work include random selection protocols arising in the study of

zero-knowledge [45, 73, 147], but none of these protocols provides the size information that is provided by SampWithSize.

4.1.4 $\mathsf{Sam}_{O(1)}$ vs. Sam_2

It is worthwhile noting that Theorem 4.1.1 for non-recursive collision finders (i.e., Sam_2), can be proven via a straightforward application of the lower-bound protocol of Goldwasser and Sipser [75] and the upper-bound protocol of [2, 56]. See Section 4.2.1 for an illustration of these easier proofs.

Various evidence suggests, however, that $\mathsf{Sam}_{O(1)}$ is more powerful than Sam_2. There is no known way to "collapse" the depth (i.e., to show that Sam_2 suffices to emulate Sam_d for $d > 2$), and under various assumptions there exist problems solvable using $\mathsf{Sam}_{O(1)}$ but not Sam_2 (for example, the average-case hardness of **SZK** [130] and constant-round parallelizable zero-knowledge proofs for **NP** [88], both imply constant-round statistically hiding commitment, but not collision-resistant hash functions). Therefore, we do not focus on the (admittedly simpler) proof of Theorem 4.1.1 for the case of Sam_2, and rather we build our machinery of SampWithSize in order to prove Theorem 4.1.1 for the case of $\mathsf{Sam}_{O(1)}$.

4.1.5 Contrast to Previous Work

Sam_d is in a sense a "canonical recursive collision finder". Similarly, one could consider a "canonical function inverter" that takes as input a circuit C and a value y and outputs a random element of $C^{-1}(y)$. Such an oracle would break all one-way functions. One could then ask whether it is possible to construct some kind of "AM-Inv" that emulates this canonical inverter. Such a result would strengthen our main theorem, since an inverter can in particular find collisions.

Unfortunately, it is not known how to build such an AM-Inv. The main difficulty is handling cheating provers, who claim that the given query is not invertible. Notice that for the problem of inverting a function, it is possible to ask queries (C, y) where y is not in the image of C.[3] In this case the oracle must say that the query is invalid. Since there is no efficient way to verify that y is *not* in the image of C, a cheating prover can claim, say, that none of the verifier's queries are in the image of C even when some are valid queries. In general, it is not known how to catch this kind of cheating, since proving that y is not in the image of C is a **co-NP** statement.

As already mentioned in Section 4.1.3, various works have gotten around this difficulty using additional restrictions either on the way the inverting oracle is called (e.g., non-adaptivity) or on the kinds of functions that the oracle inverts (e.g., one-way permutations). The main reason we are able to build AM–Sam whereas building "AM-Inv" seems out of reach, is that in our setting, unlike the inverting oracle, Sam_d can never respond "failure" to a query that passes the sanity checks (since these checks ensure that collisions always exist).

Organization

High level description of our techniques is given in Section 4.2. Notations, definitions and basic lemmas can be found in Section 4.3 (this include formal statements of the set lower and upper-bound protocols, and the uniform sampling protocol we discussed above). Our main technical contribution (Lemma 4.4.1) is given in Section 4.4, where our main result (Theorem 4.5.2) and its applications to understanding black-box reductions basing cryptography on **NP**-hardness are given in Section 4.5.

[3] Actually, as pointed out by [28], asking such queries might be very useful.

4.2 Our Techniques

In this section we overview our proof for Theorem 4.1.1. As a warmup, we start with the much simpler case of Sam_2 (i.e., $d = 2$), and then move to any constant d.

This overviews presented here assume familiarity with the lower-bound protocol of [75], the upper-bound protocol of [2], and the uniform-sampling protocol of [74] (see Section 4.3.5 for formal definitions).

4.2.1 The Case of Sam_2

Given an efficient oracle algorithm A, we construct an **AM** protocol that emulates A^{Sam_2} as follows: the protocol's high-level strategy is standard; the verifier tries to emulate the execution of A^{Sam_2} by picking random coins for the reduction A, and whenever A asks an oracle query to Sam_2, the verifier engages the prover in a protocol such that the distribution of the output is close to what Sam_2 would output, or else the verifier rejects.

Depth 1 queries: a query $(C_1, \perp)$ is answered as follows:

1. The verifier samples $x \stackrel{\mathrm{R}}{\leftarrow} \{0,1\}^n$ at random and send $y = C_1(x)$ to the prover.
2. The prover responds with $s = |C_1^{-1}(C_1(x))|$.
3. Using the lower-bound protocol of [75] and the upper-bound protocol of [2], the verify checks that $s \approx |C_1^{-1}(C_1(x))|$.[4]
4. The verifier stores (x_i, s_i) in a lookup table, and returns x.

[4] Actually, the upper-bound protocol of [2] does not give a useful upper-bound for *single* query, but only guarantees good upper-bound for most queries from a large enough set of queries. Nevertheless, the above approach can be slightly modified to go through, by choosing many x's, applying the set upper and lower-bounds protocol on each of them, and finally picking one of them at random.

Depth 2 queries: On query (C_2, x), the verifier checks that C_1 was asked before and was answered with x (if not it rejects). The it looks up the value of s_x previously stored and uses it to sample a random member of the set $\mathsf{Sib}(x)$ using the sampling lemma of [74]. It easily follows that this sample is close to uniformly distributed in $C_1^{-1}(C_1(x))$.

Assuming that the prover does not cause the verifier to reject with high probability, each query of A (or rather each adaptive round of parallel queries) is answered correctly (up to some small statistical deviation), and the proof follows.

4.2.2 The Case of $\mathsf{Sam}_{O(1)}$

We start by showing how to generalize the above approach for using protocol SampWithSize to implement protocol AM–Sam, and then give details on the implementation of protocol SampWithSize itself.

Let us start with a more precise description of Sam_d. On input $(C_1, \ldots, C_i, x)$, where $x \in \{0,1\}^n$ and each C_j is a circuit over $\{0,1\}^n$, Sam_d performs the following "sanity check": it checks that $i \leq d$, and if $i > 1$ then it also checks that it was previously queried on $(C_1, \ldots, C_{i-1}, x')$ (for some $x' \in \{0,1\}^n$) and answered with x. If any of these checks fail, Sam_d returns "failure". Otherwise, Sam_d returns a random element x' in $\mathcal{S}(C_1, \ldots, C_{i-1}, x) := \{x' \in \{0,1\}^n : \forall 1 \leq j \leq i-1, C_j(x') = C_j(x)\}$ (if $i = 1$, it returns a random $x' \in \{0,1\}^n$). Viewed differently, x' is a random collision with x for depth $i-1$ with respect to $C_1, \ldots, C_{i-1}$ (since it satisfies $C_j(x') = C_j(x)$ for every $1 \leq j \leq i-1$).

In protocol AM–Sam, the verifier chooses A's random coins at random and then emulates A^{Sam_d}, while answering each query $(C_1, \ldots, C_i, x)$ to Sam_d using the following subprotocol: the verifier first performs (using the data stored during previous executions, see below) the sanity check of Sam_d, and aborts and rejects in case any

of these tests fail. Otherwise it does the following:

In case $i = 1$: The verifier sets $\mathcal{S} = \{0,1\}^n$, $s = 2^n$, and $f = C_1$ and runs SampWithSize to get a random sample $x_1 \in \{0,1\}^n$ and an approximation

$$s_1 \approx |\{x' \in \{0,1\}^n \colon C_1(x_1) = C_1(x')\}| \,.$$

The verifier stores an entry $((C_1, x_1), s_1)$ in its memory, and returns x_1 to A as the query's answer.

In case $i > 1$: The verifier looks up the entry $((C_1, \ldots, C_{i-1}, x), s_{i-1})$ from its memory (the sanity checks guarantee that such an entry must exist, since x was the answer for a previous query $(C_1, \ldots, C_{i-1}, \cdot)$). Run SampWithSize on $\mathcal{S} = \mathcal{S}_i = \{x' \in \{0,1\}^n \colon \forall 1 \le j \le i-1,\ C_j(x') = C_j(x)\}$, $f = C_i$, and s_{i-1} in order to obtain $x_i \in \mathcal{S}$ and the approximation $s_i \approx |\{x' \in \mathcal{S} \colon C_i(x_i) = C_i(x')\}|$. As in the case $i = 1$, the verifier stores an entry $((C_1, \ldots, C_i, x_i), s_i)$ in its memory, and returns x_i.

To see that AM–Sam indeed behaves like A^{Sam_d}, we first note that Lemma 4.1.4 yields that for depth 1 queries, SampWithSize returns x_1 that is (close to) uniform in $\{0,1\}^n$, which is what Sam_d would answer. In addition, SampWithSize outputs a good approximation s_1 for $|C_1^{-1}(C_1(x_1))|$, which can be used as input for depth 2 queries to AM–Sam. Since s_1 is a good approximation, this means that a depth 2 query will be answered by SampWithSize with x_2 where x_2 is a near-uniform element of $C_1^{-1}(C_1(x_1))$, again, just as Sam_d would answer. SampWithSize also outputs a good approximation $s_2 \approx |C_2^{-1}(C(x_2))|$, which can be used for depth 3 queries, and so on.

The above is done in parallel for each of the k adaptive rounds of oracle queries. The approximation error of s_i grows as the depth increases, and from the formal

statement of Lemma 4.1.4 it follows that we can repeat the above process a constant number of times. Unfortunately, the accumulated error becomes super-polynomial for any $d = \omega(1)$.

The Protocol SampWithSize

The underlying idea behind the soundness proof of SampWithSize is to force the prover to behave "correctly", by using an accurate estimate of the average preimage size of $y \xleftarrow{R} D := f(U_{\mathcal{S}})$. Here, the average preimage size is defined as $\mu(D) := \mathrm{E}_{y \xleftarrow{R} f(U_{\mathcal{S}})}[\log |f^{-1}(y)|]$, where $f^{-1}(y) := \{x \in \mathcal{S} : f(x) = y\}$. (Note that this is the average on the log-scale; using this scale is crucial for the correctness of our protocol, see below).

The above estimate is then used to force the prover to give many tuples (y_i, s_i) such that $y_i \xleftarrow{R} D$ and most of the s_i are good approximations for $|f^{-1}(y_i)|$. We let $\mathsf{VerMean} = (\mathsf{P}_{\mathsf{VM}}, \mathsf{V}_{\mathsf{VM}})$ denote the protocol that guarantees an accurate estimate of the average preimage size, as stated in the following:

Lemma 4.2.1 (Verifying Average Preimage Size, informal)**.** *There is a constant-round public-coin protocol* VerMean *that on input $(f, \mathcal{S}, s)$, as in the statement of Lemma 4.1.4, and a real number μ', guarantees the following assuming that $s \approx |\mathcal{S}|$:*

Completeness: *if $\mu' = \mu(D)$, then the verifier accepts (when interacting with the honest prover) with high probability.*

Soundness: *if μ' is far from $\mu(D)$, then the verifier rejects (when interacting with any prover) with high probability.*

Proving SampWithSize using VerMean: we first show how to use VerMean to prove SampWithSize, then discuss how to prove VerMean in the next section. On input $\mathcal{S}$, f and s, where $s \approx |\mathcal{S}|$, the parties do the following:

1. The prover sends to the verifier a real number μ'. The parties run the VerMean protocol to verify that μ' is close to $\mu(D)$.

2. The verifier uses the sampling protocol of [74] to sample many uniform points $x_1, \ldots, x_\ell$ in $U_{\mathcal{S}}$, and sets $y_i = f(x_i)$ for all i.

3. The prover sends $s_1 = |f^{-1}(y_1)|, \ldots, s_\ell = |f^{-1}(y_\ell)|$ to the verifier. The parties engage in the [75] lower-bound to ensure that indeed $|f^{-1}(y_i)| \geq s_i$ for all i.

4. The verifier computes $\mu'' = \frac{1}{\ell}\sum_{i=1}^{\ell} \log s_i$ and checks whether $\mu' \approx \mu''$. If they are too far apart, it aborts. Otherwise, it outputs (x_i, s_i), for a random $i \in [\ell]$.

Since completeness is straightforward to observe from the definition of the protocols, in the following we focus on describing how to prove the soundness properties of the SampWithSize using VerMean. Intuitively, the lower-bound in Step 3 means that if the prover wants to cheat, it can only claim s_i to be smaller than $|f^{-1}(y_i)|$. On the other hand, by a Chernoff bound we know that for large enough ℓ, the empirical average $\frac{1}{\ell}\sum_{\ell} \log |f^{-1}(y_i)|$ will be close to $\mu(D) \approx \mu'$. Therefore, if the prover consistently under-estimates $|f^{-1}(y_i)|$ for many i, then μ'' will much smaller than μ', and we will catch him in Step 4. Together, this implies that $s_i \approx |f^{-1}(y_i)|$ for almost all i, and so outputting (x_i, s_i) for a random i is good with high probability.[5]

Verifying the Average Preimage Size

As a warmup, we first give such a verification protocol for the simple case of efficiently samplable sets. We then give an high level description of the seemingly much more complicated case of efficiently decidable sets.

[5]Here the log-scale is crucial. Assume that we would have defined $\mu(D) := \mathrm{E}_{y \xleftarrow{R} f(U_{\mathcal{S}})}[|f^{-1}(y)|]$. In this case, the standard deviation "allows" a $\Omega(2^{n/2})$ deviation from the expectation. Hence, the prover can under count the value $|f^{-1}(y)|$ for *all* the (polynomially many) samples without being caught.

Warmup: efficiently samplable sets. As already mentioned in Section 4.1.4, proving VerMean for the case $\mathcal{S} = \{0,1\}^n$, or more generally for an efficiently samplable $\mathcal{S}$, is a straightforward application of the lower-bound protocol of [75] and the upper-bound protocol of [2, 56]). In particular, the following simple protocol suffices:

1. The verifier samples many uniform random samples $x_1, \ldots, x_\ell$ from $\mathcal{S}$, computes $y_i = f(x_i)$ for all i, and he sends $y_1, \ldots, y_\ell$ to the prover.

2. The prover responds with $s_1, \ldots, s_\ell$, where $s_i = |f^{-1}(y_i)|$.

3. In parallel for all $i \in [\ell]$, the verifier engages the prover in the set lower and upper-bound protocols to check that $s_i \approx |f^{-1}(y_i)|$. Notice that the verifier is able to run the upper-bound protocol because he has the secret sample x_i.

4. If all the checks pass, then the verifier accepts iff μ' is very close to $\frac{1}{\ell}\sum_{i=1}^{\ell} \log s_i$.

By a Chernoff bound we know that $\mu(D) \approx \frac{1}{\ell}\sum_{i=1}^{\ell} \log |f^{-1}(y_i)|$ with high probability, and so if the verifier accepts in the upper/lower-bound protocols, it then also holds that $\mu(D) \approx \frac{1}{\ell}\sum_{i=1}^{\ell} \log s_i$ with high probability. This implies that the verifier accepts if $\mu' = \mu(D)$ and rejects if μ' is far from $\mu(D)$.

The general case: efficiently decidable sets. The above approach heavily relies on the set upper-bound protocol, which requires a random secret sample from $\mathcal{S}$. While obtaining such a sample is easy for efficiently sample sets, it seems infeasible when the set in hand in only efficiently decidable. Nevertheless, we manage handle the general case by asking the prover for more infirmation about the distribution $D = f(U_{\mathcal{S}})$. Namely, we show that one can verify the *histogram* of D (see Section 4.2.3), and from this histogram the verify compute the value $\mu(D)$ by itself. Finding a more direct protocol for estimating the mean, is an interesting open problem.

4.2.3 Histograms

Let D be a distribution over $\{0,1\}^n$, let $p_y = \Pr_{y' \xleftarrow{R} D}[y' = y]$, let $\epsilon \in (0,1]$ and let $m = n/\epsilon$. The ϵ-histogram of D is a function $h^f : \{0,\ldots,m\} \mapsto [0,1]$, where $h^f(i) = \Pr_{y \xleftarrow{R} D}\left[p_y \in (2^{-(i+1)\epsilon}, 2^{-i\epsilon}]\right]$. Namely, the histogram tells us the distribution of weights of elements drawn from D. Note that the smaller the ϵ, the more informative h^f is about D, but h^f's description is larger. Hence, we will consider histograms for small enough $\epsilon = 1/\operatorname{poly}(n)$.

In the following we define a distance between histograms with the following properties: (1) it is feasible verify whether a claimed histogram is in small distance from the real histogram (without using upper-bound protocols), (2) and given that the claimed histogram is of small distance from the real one, the mean derived by this histogram is close to the real value.

Wasserstein Distance

We use the 1st Wasserstein distance $\mathsf{W1}$ (also known as Kantorovich distance and Earth Mover's distance) as the distance measure between histograms. This distance is well studied in probability theory [103, 104, 122] and also has application in computer science (e.g., in the realm of image processing [145]). To understand this distance intuitively, think of a histogram h as piles of "earth" on the discrete interval $0,\ldots,m$, where the larger $h(i)$ is, the larger the pile at location i. $\mathsf{W1}(h,h')$ is the *minimal amount of work* that must be done to "push" the configuration of earth given by h to get the configuration given by h'. Recall that in physics, work equals force times distance. For example, pushing a mass of weight 0.1 from bin 2 to bin 3 requires work $0.1 \cdot (3-2) = 0.1$, where pushing the same mass from bin 2 to bin 4 requires

$0.1 \cdot (4-2) = 0.2$. The $\mathsf{W1}$ distance for histograms over $\{0, \ldots, m\}$ is defined as:

$$\mathsf{W1}(h, h') = \frac{1}{m} \cdot \sum_{0 \le i \le m} \Big| \sum_{0 \le j \le i} (h(j) - h'(j)) \Big|$$

The intuition is that $|\sum_{0 \le j \le i} h(j) - h'(j)|$ captures the amount of mass "pushed" from the interval $\{0 \ldots, i\}$ into the interval $\{i+1, \ldots, m\}$, and taking an integral over all these amounts together gives us the total amount moved.

We first notice that the above distance has the second property we require above (i.e., small variation in the Wasserstein distance implies small difference in the mean), and then, in Section 4.2.3, explain why is it feasible to verify that a claimed histogram is of small Wasserstein distance from the real one.

From the above definitions it follows that

$$\begin{aligned}
\mu(D) &= \sum_{y \in \mathrm{Supp}(D)} \Pr[D = y] \cdot \log |f^{-1}(y)| \\
&= \log |\mathcal{S}| - \sum_{y \in \mathrm{Supp}(D)} \Pr[D = y] \cdot \log \tfrac{|\mathcal{S}|}{|f^{-1}(y)|} \\
&= \log |\mathcal{S}| - \sum_{y \in \mathrm{Supp}(D)} \Pr[D = y] \cdot \log \tfrac{1}{\Pr[D=y]} \\
&= \log |\mathcal{S}| - \sum_{0 \le i \le m} \\
&\qquad \sum_{\log \frac{1}{\Pr[D=y]} \in (\epsilon \cdot (i-1), \epsilon \cdot i]} \Pr[D = y] \cdot \log \tfrac{1}{\Pr[D=y]} \\
&\approx \log |\mathcal{S}| - \sum_{0 \le i \le m} h^f(i) \cdot \frac{i}{m},
\end{aligned}$$

where the quality of $\approx$ is a function of ϵ, and $\mathrm{Supp}(D) := \{y \mid \Pr[D = y] > 0\}$. Given an histogram h such that $\mathsf{W1}(h^f, h)$ is small, the following says that the estimate

$\mu' = \log|\mathcal{S}| - \sum_{0\le i\le m} h(i)\cdot i$ is close to $\mu(D)$.

$$\begin{aligned}
|\mu(D) - \mu'| &\approx \frac{1}{m}\cdot\Big|\sum_{0\le i\le m}(h^f(i) - h(i))\cdot i\Big| \\
&= \frac{1}{m}\cdot\Big|\sum_{0\le i\le m}\sum_{0\le j\le i}(h^f(i) - h(i))\Big| \\
&\le \frac{1}{m}\cdot\sum_{0\le i\le m}\Big|\sum_{0\le j\le i}(h^f(i) - h(i))\Big| \\
&= \mathsf{W1}(h^f, h).
\end{aligned}$$

Verifying Histograms

The above tells us that in order to find the mean $\mu(D)$, it suffices to give a protocol that verifies that a histogram h is close to the true histogram h^f of D in $\mathsf{W1}$ distance. Such protocol has to handle not only efficiently samplable distributions $D = f(U_n)$, but also the efficiently samplable distributions with post-selection $D = f(U_{\mathcal{S}})$, where $\mathcal{S}$ is efficiently decidable, as long as the verifier is also given a good approximation of $|\mathcal{S}|$. We prove the following:

Lemma 4.2.2 (Verify histogram protocol, informal)**.** *There exists a constant-round public-coin protocol* $\mathsf{VerHist}$*, between a prover in* $\mathbf{BPP^{NP}}$ *and an efficient verifier, where the parties get as a common input an efficiently decidable set* $\mathcal{S} \subseteq \{0,1\}^n$*, an efficiently computable function* $f : \mathcal{S} \mapsto \{0,1\}^*$*, a good approximation (i.e., within* $(1 \pm \frac{1}{\mathrm{poly}(n)})$ *factor) of* $|\mathcal{S}|$ *and a claimed* ϵ*-histogram* $h : \{0,\dots,m\} \mapsto [0,1]$ *of the distribution* $D = f(U_{\mathcal{S}})$*, and has the following guarantees:*

Completeness. *If* $h = h^f$*, then the verifier (when interacting with the honest prover) accepts with high probability.*

Soundness. *If* h *is far from* h^f *in the 1st Wasserstein distance (as a function of* ϵ*),*

then the verifier (when interacting with any cheating prover) rejects with high probability.

Previous work using histograms. Previous works have used histograms to estimate set sizes, and a related protocol to VerHist appears in Goldreich et al. [74]. We emphasize that their protocol accomplishes a different task that is incomparable to ours.[6]

Proving Soundness of VerHist.

Before handling the general case, let us consider the very special case of VerHist where f that is promised to be a *regular* function over $\mathcal{S}$ (but with unknown regularity).

The case of regular functions. Assuming that f is k regular, it implies that $|f(\mathcal{S})| = s/k$, and the only non-zero element of the histogram is $h(k/s)$, which has value 1. To verify a claimed value k', the verifier does the following.

Preimage test: The parties run the lower-bound protocol of [75] to verify that $k = |f^{-1}(f(x))| \geq k'$ (here x is an arbitrary element in $\mathcal{S}$).

Image test: The parties run the lower-bound protocol to check that $s/k = |f(\mathcal{S})| \geq s/k'$.

From the guarantee of the lower-bound protocol, it follows that the preimage test prevents the prover from claiming $k' \gg k$. Similarly, the image test prevents the

[6]The protocol of [74] lower-bounds the size of a set $\mathcal{S}$ that is efficiently verifiable via a low-communication *interactive protocol*, but not efficiently decidable using a circuit. To do so, they recursively refine the histogram such that the following holds: if the prover lies about $|\mathcal{S}|$ (and gives an over-estimate), then at the base of the recursion the verifier catches the cheating by noticing that some parts of the histogram are empty. The prover, however, must claim they are non-empty in order to be consistent with previous answers.

prover from claiming $k' \ll k$, as this would make $|f(\mathcal{S})| = s/k \ll s/k'$ and the lower-bound of the image test would fail. Note that we are able to use the lower-bound protocol in the image test, since $f(U_{\mathcal{S}})$ is efficiently decidable.

The general case. The idea in the regular case above is that by giving a lower-bound on the image of the function f, one obtains an upper-bound on the preimage. This idea extends to f that are far from being regular, and we generalize the special case of regular functions to a protocol with more image tests over many *subsets* of $f(\mathcal{S})$.

Define the sets $\mathcal{W}_i \subseteq f(\mathcal{S})$ given by $\mathcal{W}_i = \{y \in f(\mathcal{S}) \colon |f^{-1}(y)| \geq i\}$. As in the case of regular functions, where the regularity k determines the size of the image $|f(\mathcal{S})|$, for the general case we observe the histogram h^f determines the sizes of $|\mathcal{W}_i|$ for all i. Let w_i^h be the estimate of $|\mathcal{W}_i|$ that is given by the histogram h. Note that using the lower-bound protocol, one can efficiently verify membership in $\mathcal{W}_i$ (given y, run the lower-bound to verify that $|f^{-1}(y)| \geq i$). Therefore, the set sizes $|\mathcal{W}_i|$ can themselves be lower-bounded using (a generalization of) the lower-bound protocol of [75].

In our VerHist protocol, given an input $(\mathcal{S}, f, s)$ and a claimed ϵ-histogram h for $D = f(U_{\mathcal{S}})$, the verifier first checks that h is a valid histogram (i.e., $\sum_{0 \leq j \leq m} h(j) = 1$), and then the parties are engaged in the following two steps:

Preimage test:

1. The parties run the uniform sampling protocol of [74], to sample ℓ random elements $y_1, \ldots, y_\ell$ from $D = f(U_{\mathcal{S}})$.
2. The prover sends $s_1 = |f^{-1}(y_1)|, \ldots, s_\ell = |f^{-1}(y_\ell)|$ to the verifier.
3. The parties run in the set lower-bound protocol of [75] to verify that $|f^{-1}(y_i)| \geq s_i$ for all i.

4. The verifier constructs the histogram h^{emp} induced by $s_1, \ldots, s_\ell$, and aborts if $\mathsf{W1}(h, h^{\mathsf{emp}})$ is too large.

Image test: For $i \in [m]$, the parties run the lower-bound protocol to verify that $|\mathcal{W}_i| \geq w_i^h$.

To see why the above protocol has the desired properties see Section 4.4.

4.3 Preliminaries

4.3.1 Notation

We use calligraphic letters to denote sets and capital letters to denote random variables. Given $u \in \mathcal{U}^{m+1}$, we denote the components of u as $u = (u_0, \ldots, u_m)$, and let $u_{\leq i} = (u_0, \ldots, u_i)$. For a random variable X, we write $x \xleftarrow{\mathrm{R}} X$ to indicate that x is selected according to X. Similarly, we write $x \xleftarrow{\mathrm{R}} \mathcal{S}$ to indicate that x is selected according to the uniform distribution over the set $\mathcal{S}$. Also recall that by $U_{\mathcal{S}}$ we denote the random variable whose distribution is uniform over $\mathcal{S}$. All the logarithms written as log in this thesis are in base 2. For any $m \in \mathbb{N}$, we let $[m] = \{1, \ldots, m\}$ and $(m) = \{0, 1, \ldots, m\}$.

4.3.2 The Histogram of a Function

For any distribution D, let $p_y = \Pr_{y' \xleftarrow{\mathrm{R}} D}[y = y']$ be the weight of an element y under the distribution D. The histogram of D is the probability distribution of weights of elements drawn from D. Namely, a histogram h assigns to every $p \in [0, 1]$ the probability $h(p) = \Pr_{y \xleftarrow{\mathrm{R}} D}[p = p_y]$. We will discretize the histogram on the log-scale with an approximation error ϵ to obtain the following definition.

Definition 4.3.1 (Histogram)**.** Let $\mathcal{S} \subseteq \{0,1\}^n$, let f be a function from $\mathcal{S}$ to $\{0,1\}^*$, let $\epsilon > 0$ and let $m = \lfloor n/\epsilon \rfloor$. For $i \in (m)$ we define the i'th "bin" as

$$\mathcal{B}_i = \left\{ y \colon \Pr_{x \xleftarrow{R} \mathcal{S}} [f(x) = y] \in (2^{-(i+1)\cdot\epsilon}, 2^{-i\epsilon}] \right\}.$$

Let $\mathsf{Bin}(x) := i$ iff $f(x) \in \mathcal{B}_i$, and let $h = (h_0, \ldots, h_m)$ where $h_i = \Pr_{x \xleftarrow{R} \mathcal{S}}[\mathsf{Bin}(x) = i]$. We call h the *ϵ-histogram of the function f over $\mathcal{S}$*.

For simplicity, in the definition of a histogram, we always assume that ϵ is chosen in a way that $n/\epsilon \in \mathbb{N}$ and $m = n/\epsilon$ exactly. Notice that the bins with smaller numbers contain "heavier" elements (namely for smaller i, the elements $y \in \mathcal{B}_i$ occur with larger probability). The histogram h encodes the (approximate) regularity structure of the function f over the domain $\mathcal{S}$. For example let $\epsilon = 1$ (which implies $m = n$) and $\mathcal{S} = \{0,1\}^n$. Therefore a 1-to-1 function's histogram has one non-zero entry $h_m = 1$, a 2-to-1 function's histogram has one non-zero entry $h_{m-1} = 1$, while a constant function's histogram has one non-zero entry $h_0 = 1$. Functions with more complex regularity structures would have more complex histograms.

Histograms can also be defined for *empirical* samples drawn according to a distribution as follows. Suppose we have a set of examples $x_1, \ldots, x_\ell$ all sampled from $x_i \xleftarrow{R} \mathcal{S}$, and suppose someone claims to us the weights of each of the $f(x_i)$. Namely, he gives to us a labeling function $v : [\ell] \to (m)$ with the claim that $f(x_i) \in \mathcal{B}_{v(i)}$. This labeling v induces a claimed histogram as follows.

Definition 4.3.2 (Empirical histogram)**.** For a labeling function $v : [\ell] \mapsto (m)$, define $h^v = \mathsf{Hist}(v)$ where $h^v_j = \Pr_{i \xleftarrow{R} [\ell]}[v(i) = j]$.

The following observations easily follow from Definition 4.3.1.

Proposition 4.3.3.

1. $\bigcup_{i\in(m)} \mathcal{B}_i = \mathcal{S}$ *and* $\sum_{i\in(m)} h_i = 1$,
2. *For every* $y \in \mathcal{B}_i$ *it holds that* $|f^{-1}(y)| \in (|\mathcal{S}| \cdot 2^{-(i+1)\epsilon}, |\mathcal{S}| \cdot 2^{i\cdot\epsilon}]$.
3. *For every* $i \in (m)$ *it holds that* $|\mathcal{B}_i| \in [h_i \cdot 2^{i\epsilon}, h_i \cdot 2^{(i+1)\cdot\epsilon})$.

4.3.3 Metrics over Distributions

Wasserstein Distance

The following metric measures how much "work" it takes to turn one distribution over $\mathcal{U}$ into another one; where the amount of work is assumed to be amount of needed "moves" of the probability masses times the distance by which they are moved. Our definition is for the special case that the members of $\mathcal{U}$ form a one-dimensional array and their distance is the normalized difference between their indices. For more general spaces it is known as the 1st Wasserstein distance or the Kantorovich distance [103, 104, 122]. Also in the field of image processing it known as the Earth Mover's distance [145].

Definition 4.3.4 (1st Wasserstein distance over arrays)**.** Given two distribution vectors x and y over (m), for every $i \in (m)$ we let $a_i = \sum_{0\le j\le i} x_j$ and $b_i = \sum_{0\le j\le i} y_j$. We let

- $\overrightarrow{\mathsf{W1}}(x,y) = \frac{1}{m} \cdot \sum_{i\in(m):\, a_i > b_i} (a_i - b_i)$,
- $\overleftarrow{\mathsf{W1}}(x,y) = \frac{1}{m} \cdot \sum_{i\in(m):\, b_i > a_i} (b_i - a_i)$,
- $\mathsf{W1}(x,y) = \overrightarrow{\mathsf{W1}}(x,y) + \overleftarrow{\mathsf{W1}}(x,y)$,

where we call $\mathsf{W1}(x,y)$ the 1st Wasserstein distance between x and y, where $\overleftarrow{\mathsf{W1}}(x,y)$ and $\overrightarrow{\mathsf{W1}}(x,y)$ are called the left and right Wasserstein distance respectively.

The triangle inequalities are easy to verify.

Proposition 4.3.5. *Let x, y and z be distributions vector over (m), then*

1. $\overrightarrow{\mathsf{W1}}(x,y) + \overrightarrow{\mathsf{W1}}(y,z) \geq \overrightarrow{\mathsf{W1}}(x,z)$,
2. $\overleftarrow{\mathsf{W1}}(x,y) + \overleftarrow{\mathsf{W1}}(y,z) \geq \overleftarrow{\mathsf{W1}}(x,z)$,
3. $\mathsf{W1}(x,y) + \mathsf{W1}(y,z) \geq \mathsf{W1}(x,z)$.

Proof Sketch: We only prove the first item. For $i \in (m)$, let a_i and b_i be as in Definition 4.3.4 and similarly let $c_i = \sum_{j \in (i)} z_j$. Let $i \in (m)$ be such that $(a_i - c_i) > 0$, we will show this coordinate contributes at least as much to $\overrightarrow{\mathsf{W1}}(x,y) + \overrightarrow{\mathsf{W1}}(y,z)$ as it does to $\overrightarrow{\mathsf{W1}}(x,z)$ (this concludes the proof, since only positive $(a_i - c_i)$ contributes to $\overrightarrow{\mathsf{W1}}(x,z)$). Assuming that both $(a_i - b_i)$ and $(b_i - c_i)$ are positive, then this coordinates contributes $\frac{1}{m} \cdot ((a_i - b_i) + (b_i - c_i)) = \frac{1}{m} \cdot (a_i - c_i)$ to $\overrightarrow{\mathsf{W1}}(x,y) + \overrightarrow{\mathsf{W1}}(y,z)$. In the case that one of $(a_i - b_i)$ and $(b_i - c_i)$ is positive and the other is negative, then the only (positive) term that contributes to $\overrightarrow{\mathsf{W1}}(x,y) + \overrightarrow{\mathsf{W1}}(y,z)$ is larger than $\frac{1}{m} \cdot (a_i - c_i)$. $\square$

Shift Distance

Suppose we have a set of empirical examples $x_1, \ldots, x_\ell \overset{\mathrm{R}}{\leftarrow} \mathcal{S}$. Let $u(i) = \mathsf{Bin}(x_i)$ and $v(i)$ be the "claimed" value for $u(i)$ (which might differ from the honest bin labels $\mathsf{Bin}(x_i) = u(i)$). Let h^v be the histogram induced by the (possibly false) bin labels $v(i)$, and let h^u be the histogram induced by the true bin labels $\mathsf{Bin}(x_i) = u(i)$. While $\mathsf{W1}(h^u, h^v)$ captures the minimal amount of work to move the histogram h^u to the histogram h^v. The following "shift distance" captures the amount of work required by a *specific* way to move from one histogram to another (and thus at least as large as $\mathsf{W1}(h^u, h^v)$).

Definition 4.3.6 (Shift distance)**.** Given two mappings u, v from $[\ell]$ to (m), we define the right shift distance as $\overrightarrow{\mathsf{SH}}(u,v) = \frac{1}{m\ell} \cdot \sum_{i\colon u(i)<v(i)}(v(i)-u(i))$, the left shift distance as $\overleftarrow{\mathsf{SH}}(u,v) = \frac{1}{m\ell} \cdot \sum_{i\colon u(i)>v(i)}(u(i)-v(i))$, and the shift distance as $\mathsf{SH}(u,v) = \overrightarrow{\mathsf{SH}}(u,v) + \overleftarrow{\mathsf{SH}}(u,v)$.

We will use the following simple proposition.

Proposition 4.3.7. *The following holds for every two mappings u and v, from $[\ell]$ to (m):*

1. *If $u(i) \le v(i)+k$ for all $i \in [\ell]$, then $\overleftarrow{\mathsf{SH}}(u,v) \le k/m$. Similarly if $v(i)-k \le u(i)$ for all $i \in [\ell]$, then $\overrightarrow{\mathsf{SH}}(u,v) \le k/m$.*

2. *If $\mathsf{SH}(u,v) \le k/m$, then for at least $(1-\delta)$ fraction of $i \in [\ell]$ it holds that $|u(i)-v(i)| \le k/\delta$.*

3. *It holds that $\overrightarrow{\mathsf{SH}}(u,v) = \frac{1}{m\ell} \cdot \sum_{j\in(m)} |\{i\colon u(i) \le j \wedge v(i) > j\}|$, and similarly*

$$\overleftarrow{\mathsf{SH}}(u,v) = \frac{1}{m\ell} \cdot \sum_{j\in(m)} |\{i\colon v(i) \le j \wedge u(i) > j\}| .$$

Proof Sketch. The first part readily follows from Definition 4.3.6. For the second part note that $\mathrm{E}_{i \xleftarrow{\mathrm{R}} [\ell]}[|u(i)-v(i)|] = m \cdot \mathsf{SH}(u,v) \le k$. So by the Markov inequality it holds that $\Pr_{i \xleftarrow{\mathrm{R}} [\ell]}[|u(i)-v(i)| > k/\delta] \le \delta$. Finally, the third part holds because if $v(i) > u(i)$, then the index i contributes $\frac{1}{m\ell} \cdot (v(i)-u(i))$ to $\overrightarrow{\mathsf{SH}}(u,v)$ while it contributes $\frac{1}{m\ell}$ to the right hand side for each j such that $u(i) \le j < v(i)$. □

Shift Distance to Wasserstein Distance

The following lemma, relates the Shift distance of two samples, to the Wasserstein distance of the two histograms induced by these samples.

Lemma 4.3.8 (Bounding $\mathsf{W1}$ with SH)**.** *The following holds for every two mappings u and v, from $[\ell]$ to (m): let $h^u = \mathsf{Hist}(u)$ and $h^v = \mathsf{Hist}(v)$, then*

1. $\overrightarrow{\mathsf{W1}}(h^u, h^v) \leq \overrightarrow{\mathsf{SH}}(u, v)$ *and* $\overleftarrow{\mathsf{W1}}(h^u, h^v) \leq \overleftarrow{\mathsf{SH}}(u, v)$ *(and thus* $\mathsf{W1}(h^u, h^v) \leq \mathsf{SH}(u, v)$*).*

2. $\overrightarrow{\mathsf{W1}}(h^u, h^v) \geq \overrightarrow{\mathsf{SH}}(u, v) - \overleftarrow{\mathsf{SH}}(u, v)$ *and* $\overleftarrow{\mathsf{W1}}(h^u, h^v) \geq \overleftarrow{\mathsf{SH}}(u, v) - \overrightarrow{\mathsf{SH}}(u, v)$ *(and thus* $\mathsf{W1}(h^u, h^v) \geq |\overrightarrow{\mathsf{SH}}(u, v) - \overleftarrow{\mathsf{SH}}(u, v)|$*).*

Proof of Lemma 4.3.8. Let $J_{u>v} := \left\{ j \in (m) \colon \sum_{i \in (j)} h_i^u > \sum_{i \in (j)} h_i^v \right\}$. By Definition 4.3.4 it holds that

$$\overrightarrow{\mathsf{W1}}(h^u, h^v) = \frac{1}{m} \cdot \sum_{j \in J_{u>v}} \left(\sum_{i \in (j)} h_i^u - \sum_{i \in (j)} h_i^v \right),$$

and by Definition 4.3.2 it holds that

$$\sum_{i \in (j)} h_i^u = \sum_{i \in (j)} \frac{1}{\ell} \cdot |\{k \colon u(k) = i\}| = \frac{1}{\ell} \cdot |\{i \colon u(i) \leq j\}| \,.$$

The first part of the lemma (we only give here the proof for $\overrightarrow{\mathsf{W1}}$, where the proof for

$\overleftarrow{\mathsf{W1}}$ is analogous) holds since

$$
\begin{aligned}
&\overrightarrow{\mathsf{W1}}(h^u, h^v) \\
&= \frac{1}{m\ell} \cdot \sum_{j \in J_{u>v}} \left(|\{i \colon u(i) \leq j\}| - |\{i \colon v(i) \leq j\}| \right) \\
&= \frac{1}{m\ell} \cdot \sum_{j \in J_{u>v}} \left(|\{i \colon u(i) \leq j \wedge v(i) > j\}| - |\{i \colon v(i) \leq j \wedge u(i) > j\}| \right) \\
&\leq \frac{1}{m\ell} \cdot \sum_{j \in J_{u>v}} |\{i \colon u(i) \leq j \wedge v(i) > j\}| \\
&\leq \frac{1}{m\ell} \cdot \sum_{j \in (m)} |\{i \colon u(i) \leq j \wedge v(i) > j\}| \\
&= \overrightarrow{\mathsf{SH}}(u, v) \qquad \text{(by Proposition 4.3.7).}
\end{aligned}
$$

The second part holds since

$$
\begin{aligned}
&\overrightarrow{\mathsf{W1}}(h^u, h^v) \\
&= \frac{1}{m\ell} \cdot \sum_{j \in J_{u>v}} \left(|\{i \colon u(i) \leq j\}| - |\{i \colon v(i) \leq j\}| \right) \\
&\geq \frac{1}{m\ell} \cdot \sum_{j \in (m)} \left(|\{i \colon u(i) \leq j\}| - |\{i \colon v(i) \leq j\}| \right) \\
&= \frac{1}{m\ell} \cdot \sum_{j \in (m)} \left(|\{i \colon u(i) \leq j \wedge v(i) > j\}| - |\{i \colon v(i) \leq j \wedge u(i) > j\}| \right) \\
&= \frac{1}{m\ell} \cdot \Big(\sum_{j \in (m)} |\{i \colon u(i) \leq j \wedge v(i) > j\}| - \sum_{j \in (m)} |\{i \colon v(i) \leq j \wedge u(i) > j\}| \Big) \\
&= \overrightarrow{\mathsf{SH}}(u, v) - \overleftarrow{\mathsf{SH}}(u, v).
\end{aligned}
$$

The last equality is due to Proposition 4.3.7. □

4.3.4 Efficient Provers for AM Protocols

In this section we show that the prover in any $\mathbf{AM}[O(1)]$ protocol can be replaced by a $\mathbf{BPP}^{\mathbf{NP}}$ prover, with a loss in the completeness error that depends on the round complexity. First we review a result of [11] about amplifying the soundness and reducing the rounds to two, in any $\mathbf{AM}[O(1)]$ protocol.

The definition above assumes that the probability a YES instance is accepted is $\geq 1 - 2^{-n}$ and the probability that a NO instance is accepted is $\leq 2^{-n}$. This is equivalent to YES instances being accepted with probability $\geq 2/3$ and NO instances being accepted with probability $\leq 1/3$ for by the following well-known lemma.

Lemma 4.3.9 (Error reduction for **AM** protocols, [11])**.** *Let* $\mathsf{M} = (\mathsf{P}, \mathsf{V})$ *be a* $\mathbf{AM}[O(1)]$ *protocol which there is no condition on the probability of accepting or rejecting the inputs by the verifier, and let* $\alpha = \alpha(n) < 1, \beta = \beta(n) < 1$ *be such that* $\alpha - \beta > 1/\operatorname{poly}(n)$*. Suppose we define the sets* $\mathcal{Y}^{\alpha}$ *and* $\mathcal{N}^{\beta}$ *as follows.*

- $\mathcal{Y}_n^{\alpha} = \{x \in \{0,1\}^n \colon \Pr[\mathsf{V} \text{ accepts in } \langle \mathsf{P}, \mathsf{V} \rangle (x)] \geq \alpha\}$.
- $\mathcal{N}_n^{\beta} = \{x \in \{0,1\}^n \colon \forall \mathsf{P}^*, \ \Pr[\mathsf{V} \text{ accepts in } \langle \mathsf{P}^*, \mathsf{V} \rangle (x)] \leq \beta\}$.

Then there exists an **AM** *protocol* $\mathsf{M}' = (\mathsf{P}', \mathsf{V}')$ *(with an efficient verifier) such that for all* n *the following holds*

- $\forall x \in \mathcal{Y}_n^{\alpha}$, $\Pr[\mathsf{V}' \text{ accepts in } \langle \mathsf{P}', \mathsf{V}' \rangle (x)] \geq 1 - 2^{-2n}$.
- $\forall x \in \mathcal{N}_n^{\beta}, \forall \mathsf{P}^*, \ \Pr[\mathsf{V}' \text{ rejects in } \langle \mathsf{P}^*, \mathsf{V}' \rangle (x)] \geq 1 - 2^{-2n}$.

The following lemma says that the prover in any $\mathbf{AM}[O(1)]$ protocol can be replaced by a $\mathbf{BPP}^{\mathbf{NP}}$ prover, with a loss in the completeness error that depends on the round complexity.

Lemma 4.3.10 ($\mathbf{BPP}^{\mathbf{NP}}$ provers for $\mathbf{AM}$ protocols)**.** *Let* $\mathsf{M} = (\mathsf{P}, \mathsf{V})$ *be an* $\mathbf{AM}[2k]$ *protocol (with no input) for* $k = O(1)$ *such that*

$$\Pr[\mathsf{V} \text{ accepts in } \langle \mathsf{P}, \mathsf{V} \rangle] \geq 1 - \delta,$$

for $\delta \geq 1/\operatorname{poly}(n)$*. Then there is a* $\mathbf{BPP}^{\mathbf{NP}}$ *prover strategy* P' *such that*

$$\Pr[\mathsf{V} \text{ accepts in } \langle \mathsf{P}', \mathsf{V} \rangle] \geq 1 - 2k\delta^{1/2^k}.$$

In order to prove prove Lemma 4.3.10 we start with the following lemma.

Lemma 4.3.11. *Let* $\mathsf{M} = (\mathsf{P}, \mathsf{V})$ *be a* $\mathbf{AM}[O(1)]$ *protocol,* $\delta \geq 1/\operatorname{poly}(n)$*, and the following set is not empty:*

$$\mathcal{Y}_n^{1-\delta} = \{x \in \{0,1\}^n \colon \Pr[\langle \mathsf{P}, \mathsf{V} \rangle (x) \text{ accepts}] \geq 1 - \delta\},$$

then there exists a $\mathbf{BPP}^{\mathbf{NP}}$ *strategy that with probability* $\geq 1 - 2^{-n/2}$ *finds an element* $x \in \{0,1\}^n$ *such that* $\Pr[\langle \mathsf{P}, \mathsf{V} \rangle (x) \text{ accepts}] \geq 1 - 2\delta$*.*

Proof. Define the set $\mathcal{N}_n^{1-2\delta} = \{x \in \{0,1\}^n \colon \Pr[\langle \mathsf{P}, \mathsf{V} \rangle (x) \text{ accepts}] \leq 1 - 2\delta\}$, and let $\mathsf{M}' = (\mathsf{P}', \mathsf{V}')$ be the "amplified" two-round protocol that Lemma 4.3.9 yields with respect to $\mathsf{M} = (\mathsf{P}, \mathsf{V})$ with parameters $\alpha = 1 - \delta$ and $\beta = 1 - 2\delta$. Thus,

- for every $x \in \mathcal{Y}_n^{1-\delta}$, it holds that $\Pr[\langle \mathsf{P}', \mathsf{V}' \rangle (x) \text{ accepts}] \geq 1 - 2^{-2n}$, and
- for every $x \in \mathcal{N}_n^{1-2\delta}$, it holds that $\Pr[\langle \mathsf{P}', \mathsf{V}' \rangle (x) \text{ accepts}] \leq 2^{-2n}$.

Let ω denote the random coins of V' for inputs of length n. By a union bound, it holds that

$$\Pr_{\omega}[\exists x \in \mathcal{N}_n^{1-2\delta},\ \langle \mathsf{P}', \mathsf{V}' \rangle (x; \omega) \text{ accepts}] \leq 2^{-n} \tag{4.3.1}$$

The $\mathbf{BPP}^{\mathbf{NP}}$ algorithm claimed in the theorem simply does the following: choose ω uniformly at random, and use the $\mathbf{NP}$ oracle to find x such that $\mathsf{V}'(x;\omega) = 1$. Notice that since $\mathcal{Y}_n^{1-\delta} \neq \varnothing$, therefore with probability at least $1 - 2^{-2n}$ such x exists. Furthermore, by Inequality 4.3.1 the probability that $x \in \mathcal{N}_n^{1-\delta}$ is at most 2^{-n}.

Therefore, with probability $1 - 2^{-n} - 2^{-2n} \geq 1 - 2^{-n/2}$ we output $x \notin \mathcal{N}_n^{1-\delta}$, namely it will hold that $\Pr[\langle \mathsf{P}, \mathsf{V} \rangle (x) \text{ accepts}] \geq 1 - 2\delta$. □

Proof of Lemma 4.3.10. For any sequence of i messages $w = (r_1, m_1, r_2, \dots)$ exchanged between V and P we can always define a $\mathbf{AM}[k-i]$ game $(\mathsf{P}^w, \mathsf{V}^w)$ with respect to w where the fist i messages are *fixed* to be w and the parties continue interacting as if they are (P, V). For any sequence of messages w we define $\rho(w) = \Pr[\mathsf{V}^w \text{ accepts in } \langle \mathsf{P}^w, \mathsf{V}^w \rangle]$.

Suppose r_1 is the first message of the verifier V. By an average argument it holds that $\Pr[\rho(r_1) \geq 1 - \sqrt{\delta}] \geq 1 - 2\sqrt{\delta}$. The prover strategy P' pretends that $\rho(r_1) \geq 1 - \sqrt{\delta}$ holds and uses Lemma 4.3.11 where the input x of Lemma 4.3.11 will be the response of P' to the message r_1. Lemma 4.3.11 yields that if $\rho(r_1) \geq 1 - \sqrt{\delta}$ then with probability $1 - \text{neg}(n)$, P' finds a message $x = m_1$ such that $\rho(r_1, m_1) \geq 1 - 2\sqrt{\delta}$. Inductively if $\rho(r_1, m_1, \dots, r_{i-1}, m_{i-1}) \geq 1 - 2\delta^{1/2^i}$, then it holds that

$$\Pr[\rho(r_1, m_1, \dots, r_{i-1}, m_{i-1}, r_i) \geq 1 - \delta^{1/2^{i+1}}] \geq 1 - 2\delta^{1/2^{i+1}}$$

and if $\rho(r_1, m_1, \dots, r_{i-1}, m_{i-1}, r_i) \geq 1 - \delta^{1/2^{i+1}}$, then the prover can use the $\mathbf{BPP}^{\mathbf{NP}}$ strategy of Lemma 4.3.11 to find m_i such that $\rho(r_1, m_1, \dots, r_{i-1}, m_{i-1}, r_i, m_i) \geq 1 - 2\delta^{1/2^{i+1}}$. If at the end P' achieves $\rho(r_1, \dots, m_k) > 0$ he succeeds. By a union bound the latter happens with probability at least $1 - 2\sum_{i \in [k]} \delta^{1/2^i} > 1 - 2k\delta^{1/2^k}$.

□

4.3.5 Set Size Estimation and Sampling Protocols

We call a family of sets $\{\mathcal{S}_n\}$ (non-uniformly) efficiently decidable, if there exist a family of polynomial size circuits Boolean $\{C_n\}$ such that $\mathcal{S}_n = \{x \mid C_n(x) = 1\}$. When it is clear from the context we simply write $\mathcal{S}$ instead of $\mathcal{S}_n$.

The following fundamental lemma by [75] provides a protocol to lower-bound the size of efficiently decidable sets up to small multiplicative factor.

Lemma 4.3.12 (Set lower-bound protocol [75])**.** *There exists an* **AM** *protocol* $\mathsf{SetLB} = (\mathsf{P_{LB}}, \mathsf{V_{LB}})$*, where the parties get as input an efficiently decidable set* $\mathcal{S} \subseteq \{0,1\}^n$*,*[7] s *(as size of* $\mathcal{S}$*),* ϵ *(as the approximation parameter), the verifier runs in time* $\mathrm{poly}(n, 1/\epsilon)$ *and the following holds.*

Completeness. *If* $|\mathcal{S}| \geq s$*, then* $\Pr[\mathsf{V_{LB}}$ *accepts in* $\langle \mathsf{P_{LB}}, \mathsf{V_{LB}} \rangle (\mathcal{S}, s, \epsilon)] \geq 1 - 2^{-n}$*.*

Soundness. *If* $|\mathcal{S}| \leq (1-\epsilon) \cdot s$*, then for every prover* P^**, it holds that*

$\Pr[\mathsf{V_{LB}}$ *accepts in* $\langle \mathsf{P}^*, \mathsf{V_{LB}} \rangle (\mathcal{S}, s, \epsilon)] < 2^{-n}$*.*

We will need a variation of the protocol of Lemma 4.3.12 over the promise languages. In fact the exact same protocol given in [75] (with the help of the amplification of Lemma 4.3.9) proves *both* Lemma 4.3.12 and the following Lemma 4.3.13. The protocol of [75] is described for an **AM** set $\mathcal{S}$ which clearly contain the efficiently decidable $\mathcal{S}$'s (i.e. Lemma 4.3.12) as a special case. However the same protocol (and in fact even the same analysis) given in [75], when considered over **AM** *promise* languages yields the following.

Lemma 4.3.13 (Generalized set lower-bound protocol [75])**.** *Let* $\mathsf{M} = (\mathsf{P}, \mathsf{V})$ *be an* **AM** *protocol with YES instances* $\{\mathcal{Y}_n\}$ *and non-promise instances* $\{\mathcal{T}_n\}$*. Then there*

[7] $\mathcal{S}$ can be represented as its deciding circuit, or in case of more succinct representations $(\mathsf{P_{LB}}, \mathsf{V_{LB}})$ can directly depend on the deciding algorithm of $\mathcal{S}$ when it is uniform.

exists an **AM** *protocol* $\mathsf{GenSetLB} = (\mathsf{P}_{\mathsf{GLB}}, \mathsf{V}_{\mathsf{GLB}})$*, where the parties get as input* s *(as size of* $\mathcal{Y}_n$*),* ϵ *(as the approximation parameter), the verifier runs in time* $\mathrm{poly}(n/\epsilon)$ *and the following holds.*

Completeness. *If* $|\mathcal{Y}_n| \geq s$*, then* $\Pr[\mathsf{V}_{\mathsf{GLB}}$ *accepts in* $\langle \mathsf{P}_{\mathsf{GLB}}, \mathsf{V}_{\mathsf{GLB}} \rangle (\mathsf{M}, s, \epsilon)] \geq 1 - 2^{-n}$*.*

Soundness. *If* $|\mathcal{Y}_n \cup \mathcal{T}_n| \leq (1-\epsilon) \cdot s$*, then for every prover* P^**, it holds that* $\Pr[\mathsf{V}_{\mathsf{GLB}}$ *accepts in* $\langle \mathsf{P}^*, \mathsf{V}_{\mathsf{GLB}} \rangle (\mathsf{M}, s, \epsilon)] < 2^{-n}$*.*[8]

The following lemma and its proof are an adaptation of those of [74, Lemma A.2] for the setting where we are only given an approximation of the size of the set to be sampled from. ([5] also describes a candidate protocol without the proof.)

Lemma 4.3.14 (Uniform sampling protocol, [5, 74])**.** *There exists an* **AM** *protocol* $\mathsf{UnifSamp}$*, between an* $\mathbf{P}^{\mathbf{NP}}$ *prover* P_{US} *and an efficient verifier* V_{US}*, whose parties get as input, an efficiently decidable set* $\mathcal{S} \subseteq \{0,1\}^n$*,* $s \in \mathbb{N}$*, and an error parameter* $\delta < 1$*, such that the following holds: assuming that* $s \in [(1 \pm \delta/32) \cdot |\mathcal{S}|]$*, then the verifier runs in* $\mathrm{poly}(n, 1/\delta)$ *and either rejects by outputting* $x = \perp$ *or outputs an element* $x \in \mathcal{S}$ *such that:*

Completeness. V_{US} *rejects in* $\langle \mathsf{P}_{\mathsf{US}}, \mathsf{V}_{\mathsf{US}} \rangle (\mathcal{S}, s, \delta)$ *with probability at most* δ*.*

Soundness. $\Delta(x, U_{\mathcal{S}}) \leq \Pr[\mathsf{V}_{\mathsf{US}}$ *rejects in* $\langle \mathsf{P}^*, \mathsf{V}_{\mathsf{US}} \rangle (\mathcal{S}, s, \delta)] + \delta$*, for any (unbounded) prover* P^**.*

Since the full proof of Lemma 4.3.14 is not available in the literature, here we give a proof for completeness.

[8] Note that M is not really an extra input and the protocol $\mathsf{GenSetLB}$ implicitly depends on M, but we still indicate it as so for the sake of clarity.

Proof of Lemma 4.3.14.

Protocol 4.3.15. *Set* $k = \log(5/\delta)$ *and* $\ell = \left\lfloor \log(\frac{(\delta/5)^3 s}{2k^2}) \right\rfloor$. *Let* $H_{n,\ell}$ *be an efficient family of* $2k$*-wise independent hash functions mapping* n *bits to* ℓ *bits. Set* $t = \frac{1-\delta/5}{1+\delta/32} \cdot \frac{s}{2^\ell}$.

1. $\mathsf{V_{US}}$ *picks* $h \xleftarrow{R} H_{n,\ell}$ *and sends* h *to* $\mathsf{P_{US}}$.

2. $\mathsf{P_{US}}$ *computes distinct* $x_1, \ldots, x_t \in \mathcal{S} \cap h^{-1}(0)$ *and sends them to the verifier (or aborts if such* $x_1, \ldots, x_t$ *do not exist).*

3. $\mathsf{V_{US}}$ *checks that she has received distinct* $x_1, \ldots, x_t$ *and that* $x_i \in \mathcal{S} \cap h^{-1}(0)$ *for all* $i \in [t]$ *and reject if any of the checks does not hold. If not rejected, pick* $i \xleftarrow{R} [t]$ *and output* x_i.

Completeness. The only case where the honest prover might cause the verifier to abort is if there do not exist t distinct elements in $|\mathcal{S} \cap h^{-1}(0)|$, which we call the bad event W. For every $x \in \mathcal{S}$ define ζ_x to be the random variable that is 1 if $h(x) = 0$

and zero otherwise, and let $\overline{\zeta_x} = \zeta_x - 2^{-\ell}$ such that $\mathrm{E}[\overline{\zeta_x}] = 0$. We derive:

$$\begin{aligned}
\Pr[W] &= \Pr[\sum_{x\in\mathcal{S}} \zeta_x < t] \\
&= \Pr[\sum_{x\in\mathcal{S}} \zeta_x < \tfrac{1-\delta/5}{1+\delta/32} \cdot \tfrac{s}{2^\ell}] \\
&\le \Pr[\sum_{x\in\mathcal{S}} \zeta_x < (1-\delta/5)|\mathcal{S}|/2^\ell] && \text{by the promise } s \in [(1\pm\delta/32)|\mathcal{S}|] \\
&\le \Pr\left[\sum_{x\in\mathcal{S}} \overline{\zeta_x} < -(\delta/5)|\mathcal{S}|2^{-\ell}\right] \\
&\le \frac{\mathrm{E}\left[\left(\sum_{x\in\mathcal{S}} \overline{\zeta_x}\right)^{2k}\right]}{((\delta/5)|\mathcal{S}|2^{-\ell})^{2k}} && \text{raise to } 2k \text{ power and use Markov} \\
&= \frac{\mathrm{E}\left[\sum_{x_1,\ldots x_{2k}\in\mathcal{S}} \prod_{i=1}^{2k} \overline{\zeta_{x_i}}\right]}{(\delta/5)^{2k}|\mathcal{S}|^{2k}2^{-2k\ell}} \\
&= \frac{\sum_{x_1,\ldots x_{2k}\in\mathcal{S}} \mathrm{E}\left[\prod_{i=1}^{2k} \overline{\zeta_{x_i}}\right]}{(\delta/5)^{2k}|\mathcal{S}|^{2k}2^{-2k\ell}}.
\end{aligned}$$

By $2k$-wise independence, all terms in the numerator of the last expression above given by tuples $(x_1, \ldots, x_{2k})$ where one of the x_i is unique will contribute 0 to the sum. Therefore it suffices to count such tuples where no x_i is unique.

Claim 4.3.16.

$$\sum_{x_1,\ldots,x_{2k}\in\mathcal{S}} \mathrm{E}\left[\prod_{i=1}^{2k} \overline{\zeta_{x_i}}\right] \le |\mathcal{S}|^k k^{2k} 2^{-k\ell}.$$

By Claim 4.3.16 we conclude that

$$\Pr[W] \le \frac{|\mathcal{S}|^k k^{2k} 2^{-k\ell}}{(\delta/5)^{2k}|\mathcal{S}|^{2k}2^{-2k\ell}} \le \left(\frac{k^2 2^\ell}{(\delta/5)^2|\mathcal{S}|}\right)^k \le (\delta/10)^k < \delta,$$

as claimed.

Proof. (of Claim 4.3.16) There are $\binom{|\mathcal{S}|}{i}$ ways of choosing i elements out of $|\mathcal{S}|$, and

there are at most i^{2k} ways of arranging these elements (with duplicates) when there are $2k$ total elements. For all $c \geq 2$, the expectation $\mathrm{E}[(\overline{\zeta_x})^c] \leq 2^{-\ell}$. Therefore, the sum of expectations over all tuples with exactly i non-unique elements is bounded by $\leq \binom{|\mathcal{S}|}{i} i^{2k} 2^{-i\ell}$. The entire sum is bounded by $\sum_{i=1}^{k} \binom{|\mathcal{S}|}{i} i^{2k} 2^{-i\ell}$ (we do not sum $i > k$ as these terms have duplicate x_i's and so therefore their expectation is 0). By Stirling's approximation the maximum term is for $i = k$, therefore the sum of all elements is bounded by $k\binom{|\mathcal{S}|}{k} \cdot k^{2k} 2^{-k\ell} \leq |\mathcal{S}|^k \cdot k^{2k} 2^{-k\ell}$. □

Soundness. Since a random sample from $\mathcal{S}$ is never the failure symbol $\perp$ while the protocol might output $\perp$, it follows that

$$\begin{aligned}\Delta(x, \mathcal{S}) &= \max_{T \subseteq \mathcal{S} \cup \{\perp\}} \{\Pr[x \in T] - \Pr[\mathcal{S} \in T]\} \\ &= \Pr[x = \perp] + \max_{T \subseteq \mathcal{S}} \{\Pr[x \in T] - \Pr[\mathcal{S} \in T]\}.\end{aligned}$$

Therefore, it suffices (and is actually equivalent) to show that for all $T \subseteq \mathcal{S}$, it holds that

$$\Pr[x \in T] \leq \frac{|T|}{|\mathcal{S}|} + \delta.$$

Now fix any $T \subseteq \mathcal{S}$, and if $|T| < (\delta/4)|\mathcal{S}|$ then simply pad T with extra elements until we have $|T| = (\delta/4)|S|$. Let W_T be the event that $|h^{-1}(0) \cap T| > (1 + \delta/5)2^{-\ell}|T|$

elements. Let $\zeta_x, \overline{\zeta_x}$ be defined as in the case of completeness.

$$\begin{aligned}
\Pr[W_T] &= \Pr[\sum_{x\in T} \zeta_x > (1+\delta/5)2^{-\ell}|T|] \\
&= \Pr[\sum_{x\in T} \overline{\zeta_x} > (\delta/5)2^{-\ell}|T|] \\
&\leq \frac{\sum_{x_1,\ldots,x_{2k}} \mathrm{E}[\prod_{i=1}^{2k} \overline{\zeta_{x_i}}]}{(\delta/5)^{2k}2^{-2k\ell}|T|^{2k}} \\
&\leq \left(\frac{k^2 2^\ell}{(\delta/5)^2|T|}\right)^k
\end{aligned}$$

Using the assumption $|T| \geq (\delta/5)|\mathcal{S}|$ and the definition of ℓ, k, we have that $\frac{k^2 2^\ell}{(\delta/5)^2|T|} \leq 1/2$ and so the probability that W_T occurs is $\leq \delta/5$. Assuming W_T did not occur, we have that

$$\begin{aligned}
|T \cap h^{-1}(0)| &\leq (1+\delta/5)2^{-\ell}|T| \\
&= \frac{(1+\delta/5)(1+\delta/32)}{1-\delta/5} \cdot \frac{|T|}{s} \cdot t \\
&\leq \frac{(1+\delta/5)(1+\delta/32)}{(1-\delta/5)(1-\delta/32)} \cdot \frac{|T|}{|\mathcal{S}|} \cdot t \\
&\leq (1+\delta/2) \cdot \frac{|T|}{|\mathcal{S}|} \cdot t.
\end{aligned}$$

Therefore the probability of sampling an element of T is $\delta/4 + (1+\delta/2)\frac{|T|}{|\mathcal{S}|}$, where the first term is the probability of the bad event W_T occurring while the second is the probability that conditioned on the bad event not occurring, we sample an element of T. If T was not padded then it holds that

$$\delta/4 + (1+\delta/2)\tfrac{|T|}{|\mathcal{S}|} \leq \tfrac{|T|}{|\mathcal{S}|} + \delta/2(1/2 + \tfrac{|T|}{|\mathcal{S}|}) \leq \tfrac{|T|}{|\mathcal{S}|} + 3\delta/4.$$

In case T was padded, we again get that the probability is at most

$$\delta/4 + (1+\delta/2)\delta/4 \leq 3\delta/4 \leq \frac{|T|}{|\mathcal{S}|} + 3\delta/4$$

□

Lemma 4.3.17 (Set upper-bound protocol[2])**.** *There exists a two-round public-coin protocol* $\mathsf{SetUB} = (\mathsf{P}_{\mathsf{SUB}}, \mathsf{V}_{\mathsf{SUB}})$*, where the parties get as input an efficiently decidable set* $\mathcal{S} \subseteq \{0,1\}^n$*,* s *(as size of* $\mathcal{S}$*),* $\delta, \epsilon\, 0$*, the verifier gets in addition a* secret *random sample* x *from* $\mathcal{S}$ *(unknown to the prover) and runs in time* $\mathrm{poly}(n, 1/\delta, 1/\epsilon)$*, and the following hold:*

Completeness. *If* $|\mathcal{S}| \leq s$*, then* $\Pr[\mathsf{V}_{\mathsf{SUB}}$ *accepts in* $\langle \mathsf{P}_{\mathsf{SUB}}, \mathsf{V}_{\mathsf{SUB}}(x)\rangle (\mathcal{S}, s, \delta, \epsilon)] \geq 1 - \delta$*.*

Soundness. *If* $|\mathcal{S}| \geq s(1+\epsilon)$*, then for every prover* P^**, it holds that*

$\Pr[\mathsf{V}_{\mathsf{SUB}}$ *accepts in* $\langle \mathsf{P}^*, \mathsf{V}_{\mathsf{SUB}}\rangle (\mathcal{S}, s, \delta, \epsilon)] < 1 - \epsilon/5 + \delta$*.*

4.4 Sampling with Size and Verifying The Histogram

In this section we formally state and prove Lemma 4.1.4 (SampWithSize), its extension to the multi-query case (see discussion below) and Lemma 4.2.2 (VerHist). We first prove a "weak" version of Lemma 4.2.2 (see Lemma 4.4.5), use this weak version for proving Lemma 4.1.4 and its multi-query case (formally stated as Lemma 4.4.1 and Lemma 4.4.2), and then prove Lemma 4.2.2 (stated as Lemma 4.4.4) using Lemma 4.4.2. In order to keep the text simple, we state the lemmas with respect to **PSPACE** provers, and only give a variant of Lemma 4.4.2 (Corollary 4.4.3) with

a $\mathbf{BPP}^{\mathbf{NP}}$ prover. The very same approach, however, can be applied to give similar variant of all other lemmas in this section. Let us start with the formal statements of the main lemmas and of Corollary 4.4.3.

Lemma 4.4.1 (Restating Lemma 4.1.4)**.** *There exists an* $\mathbf{AM}[O(1)]$ *protocol* $\mathsf{SampWithSize} = (\mathsf{P}_{\mathsf{sws}}, \mathsf{V}_{\mathsf{sws}})$*, whose parties get as input an efficiently decidable set* $\mathcal{S} \subseteq \{0,1\}^n$*, an accuracy parameter* $\delta \geq 1/\operatorname{poly}(n)$*,* $s \in \mathbb{N}$ *(as size of* $\mathcal{S}$*) and an efficiently computable function* $f : \mathcal{S} \mapsto \{0,1\}^*$*. At the end of the protocol* $\mathsf{V}_{\mathsf{sws}}$ *either rejects (signified by outputting a special symbol* $x = \perp$*), or it outputs a pair* (x, s_x)*. Assuming that* $s \in [(1 \pm \gamma) \cdot |\mathcal{S}|]$ *for* $\gamma := (\frac{\delta}{10n})^8$*, then the following conditions hold:*

Completeness: $\mathsf{V}_{\mathsf{sws}}$ *accepts when interacting with* $\mathsf{P}_{\mathsf{sws}}$ *with probability at least* $1 - \delta$*, and if not rejecting it holds that* $s_x = |f^{-1}(f(x))|$*.*

Soundness: *For all provers* P^**, the following hold:*

- $\Pr[\mathsf{V}_{\mathsf{sws}}$ *does not reject in* $\langle \mathsf{P}^*, \mathsf{V}_{\mathsf{sws}} \rangle \wedge s_x \notin [(1 \pm \delta) \cdot |f^{-1}(f(x))|]] \leq \delta$*,*
- $\Delta(x, U_{\mathcal{S}}) \leq \delta + \Pr[\mathsf{V}_{\mathsf{sws}}$ *rejects in* $\langle \mathsf{P}^*, \mathsf{V}_{\mathsf{sws}} \rangle]$*.*

For proving Theorem 4.1.1 we need to handle multiple queries to protocol $\mathsf{SampWithSize}$ simultaneously, using a single constant round protocol, while enforcing independence between the verifiers' outputs in the different calls. While applying Lemma 4.4.1 independently in parallel for each of the queries does not guarantee such independence (nothing prevents outputs of the different calls to be dependant), a bit more careful usage of this lemma does yield the desired guarantee, which is formally state as the next lemma.

Lemma 4.4.2 (Multi-query variant of Lemma 4.4.1)**.** *There exists an* $\mathbf{AM}[O(1)]$ *protocol* $\mathsf{MultSampWithSize} = (\mathsf{P}_{\mathsf{MSWS}}, \mathsf{V}_{\mathsf{MSWS}})$*, whose parties get as input a tuple of* k *triplets* $((\mathcal{S}_1, s_1, f_1), \ldots, (\mathcal{S}_k, s_k, f_k))$*, where each* $\mathcal{S}_i \subseteq \{0,1\}^{n_i}$ *is efficiently decidable,*

an accuracy parameter $\delta \geq 1/\operatorname{poly}(n)$ *for* $n = \sum_i n_i$, $s_i \in \mathbb{N}$ *and each* $f_i : \mathcal{S}_i \mapsto \{0,1\}^*$ *is an efficiently computable function. At end of the protocol* $\mathsf{V}_{\mathsf{MSWS}}$ *either rejects (signified by outputting a special symbol* $\perp$*) or outputs* $((x_1, s_{x_1}), \ldots, (x_k,))$*. Assuming that* $s_i \in [(1 \pm \gamma) \cdot |\mathcal{S}_i|]$ *for every* $i \in [k]$*, where* $\gamma := \frac{1}{8k} \cdot (\frac{\delta}{20n})^8$*, then the following holds:*

Completeness: $\mathsf{V}_{\mathsf{MSWS}}$ *accepts when interacting with* $\mathsf{P}_{\mathsf{MSWS}}$ *with probability at least* $1 - \delta$*, and if not rejecting it holds that* $s_{x_i} = \left|f_i^{-1}(f_i(x_i))\right|$ *for every* $i \in [k]$*.*

Soundness: *The following holds for any (unbounded) prover* P^**:*

- $\Pr\left[\langle \mathsf{P}^*, \mathsf{V}_{\mathsf{MSWS}} \rangle \neq \textit{ reject } \wedge \exists i \in [k] \colon s_{x_i} \notin [(1 \pm \delta) \cdot \left|f_i^{-1}(f_i(x_i))\right|]\right] < \delta$,
- $\mathsf{SD}((U_{\mathcal{S}_1}, \ldots, U_{\mathcal{S}_k}), (x_1, \ldots, x_k)) \leq \delta + \Pr[\mathsf{V}_{\mathsf{MSWS}} \textit{ rejects in } \langle \mathsf{P}^*, \mathsf{V}_{\mathsf{MSWS}} \rangle]$.

Corollary 4.4.3 (Lemma 4.4.2 with $\mathbf{BPP^{NP}}$ prover)**.** *Let* $((\mathcal{S}_1, s_1, f_1), \ldots, (\mathcal{S}_k, s_k, f_k))$ *and* δ *be as in Lemma 4.4.2. There exists an* $\mathbf{AM}[O(1)]$ *protocol* $\mathsf{MultSampWithSize} = (\mathsf{P}_{\mathsf{MSWS}}, \mathsf{V}_{\mathsf{MSWS}})$*, where* $\mathsf{P}_{\mathsf{MSWS}}$ *in* $\mathbf{BPP^{NP}}$*, whose parties get as input* $((\mathcal{S}_1, s_1, f_1), \ldots, (\mathcal{S}_k, s_k, f_k))$ *and* δ*, at end of the protocol* $\mathsf{V}_{\mathsf{MSWS}}$ *either rejects or outputs* $((x_1, s'_1), \ldots, (x_k, s'_k))$*, and the following holds for a universal constant* $c > 1$*. Assuming that* $s_i \in [(1 \pm \gamma) \cdot |\mathcal{S}_i|]$ *for every* $i \in [k]$*, where* $\gamma := \frac{\delta^c}{ckn^8}$*, then*

Completeness: $\mathsf{V}_{\mathsf{MSWS}}$ *accepts when interacting with* $\mathsf{P}_{\mathsf{MSWS}}$ *with probability at least* $1 - \delta$*.*

Soundness: *The following holds for any (unbounded) prover* P^**:*

- $\Pr\left[\langle \mathsf{P}^*, \mathsf{V}_{\mathsf{MSWS}} \rangle \neq \textit{ reject } \wedge \exists i \in [k] \colon s_{x_i} \notin [(1 \pm \delta) \cdot \left|f_i^{-1}(f_i(x_i))\right|]\right] < \delta$,
- $\mathsf{SD}((U_{\mathcal{S}_1}, \ldots, U_{\mathcal{S}_k}), (x_1, \ldots, x_k)) \leq \delta + \Pr[\mathsf{V}_{\mathsf{MSWS}} \textit{ rejects in } \langle \mathsf{P}^*, \mathsf{V}_{\mathsf{MSWS}} \rangle]$.

Proof. Let c' be the number of rounds of Protocol $\mathsf{MultSampWithSize}$ of Lemma 4.4.1, and let δ' be such that $\delta = 2c'\delta'^{1/2^{c'}}$. Lemma 4.4.1 yields that if $\gamma \leq \frac{1}{8k} \cdot (\frac{\delta'}{20n})^8$ then

both the completeness and the soundness conditions hold with a **PSPACE** prover and the accuracy parameter δ'. Lemma 4.3.10 yields that we can get a $\mathbf{BPP}^{\mathbf{NP}}$ prover that makes the verifier accept with probability $1 - 2c'\delta'^{1/2^{c'}} = 1 - \delta$. Therefore the honest prover can be implemented in $\mathbf{BPP}^{\mathbf{NP}}$ if γ is small enough:

$$\gamma \leq \frac{1}{8k} \cdot (\frac{\delta'}{200n})^8 = \frac{1}{8k} \cdot \left(\frac{(\delta/2c')^{2^{c'}}}{20n}\right)^8$$

which is implied by $\gamma \leq \frac{\delta^c}{ckn^8}$ for a large enough constant c. □

Lemma 4.4.4 (Restating Lemma 4.2.2)**.** *There exists an* $\mathbf{AM}[O(1)]$ *protocol* $\mathsf{VerHist} = (\mathsf{P_{VH}}, \mathsf{V_{VH}})$*, whose parties get as input an efficiently decidable set* $\mathcal{S} \subseteq \{0,1\}^n$*,* $s \in \mathbb{N}$ *(as size of* $\mathcal{S}$*), an efficiently computable function* $f : \mathcal{S} \mapsto \{0,1\}^*$*, the histogram parameter* $0 < \epsilon < 1$ *and a (claimed)* ϵ*-histogram* $h \in [0,1]^{m+1}$ *for* $m = n/\epsilon$ *such that the following holds. Let* $h^f \in [0,1]^{m+1}$ *be the (real)* ϵ*-histogram of* f *with respect to* $\mathcal{S}$*. If* $s \in [(1 \pm \gamma) \cdot |\mathcal{S}|]$ *for* $\gamma = (\frac{\epsilon}{10n})^{40}$ *then:*

Completeness: *If* $h = h^f$*, then* $\mathsf{V_{VH}}$ *accepts when interacting with* $\mathsf{P_{VH}}$ *with probability at least* $1 - 2^{-n}$*.*

Soundness: *If* $\mathsf{W1}(h^f, h) > 20/m = 20\epsilon/n$*, then (for any unbounded prover)* $\mathsf{V_{VH}}$ *rejects with probability at least* $1 - 2^{-n}$*.*

We are also using the following "supporting lemmas" whose proofs are given in Section 4.4.1.

Supporting Lemmas

The following protocol is similar to protocol $\mathsf{VerHist}$ of Lemma 4.4.4, but provides a weaker guarantee: the prover cannot convince the verifier that $\overrightarrow{\mathsf{W1}}$ is much larger than $\overleftarrow{\mathsf{W1}}$.

Lemma 4.4.5 (Weak Verify Histogram)**.** *There exists an* $\mathbf{AM}[O(1)]$ *protocol* $\mathsf{WeakVerHist} = (\mathsf{P}_{\mathsf{WVH}}, \mathsf{V}_{\mathsf{WVH}})$*, whose parties get as input an efficiently decidable set* $\mathcal{S} \subseteq \{0,1\}^n$*,* $s \in \mathbb{N}$ *(as size of* $\mathcal{S}$*), an efficiently computable function* $f : \mathcal{S} \mapsto \{0,1\}^*$*, the histogram parameter* $\epsilon = 1/\operatorname{poly}(n)$ *and a (claimed) histogram* $h \in [0,1]^{m+1}$ *for* $m = n/\epsilon$ *such that the following holds. Let* $h^f \in [0,1]^{m+1}$ *be the true* ϵ*-histogram of* f *with respect to* $\mathcal{S}$*. Given the promise that* $s \in [(1 \pm \epsilon)|\mathcal{S}|]$ *the following holds.*

Completeness: *If* $h = h^f$*, then (when interacting with* $\mathsf{P}_{\mathsf{WVH}}$*)* $\mathsf{V}_{\mathsf{WVH}}$ *accepts with probability at least* $1 - 2^{-n/2}$*.*

Soundness: *If* $\overrightarrow{\mathsf{W1}}(h^f, h) > 4 \cdot \overleftarrow{\mathsf{W1}}(h^f, h) + 100\epsilon$*, then (when interacting with any unbounded prover)* $\mathsf{V}_{\mathsf{WVH}}$ *rejects with probability at least* $1 - 2^{-n/2}$*.*

The following lemma states that there exists a protocol that verifies that a claimed empirical labeling is close to a true labeling in the shift distance.

Lemma 4.4.6 (Verify Empirical Labeling)**.** *There exists an* $\mathbf{AM}[O(1)]$ *protocol* $\mathsf{VerEmpLabel} = (\mathsf{P}_{\mathsf{VEL}}, \mathsf{V}_{\mathsf{VEL}})$*, whose parties get as input an efficiently decidable set* $\mathcal{S} \subseteq \{0,1\}^n$ *(where* $n \geq 20$*), an efficiently computable function* $f : \mathcal{S} \mapsto \{0,1\}^*$*, an histogram parameter* $\epsilon = 1/\operatorname{poly}(n)$ *and a (claimed) histogram* $h \in [0,1]^{m+1}$*. In addition, the protocol takes as input empirical samples* $x_1, \ldots, x_\ell \in \mathcal{S}$ *along with a (claimed) labeling* $u : [\ell] \mapsto (m)$*. Let* h^f *be the (true) histogram of* f *over the set* $\mathcal{S}$*. Let* u^f *be the (true) labeling of* x_i *(i.e.,* $u^f(i) = \mathsf{Bin}(x_i)$*) and let* $h^{u^f} = \mathsf{Hist}(u^f)$ *(i.e., the induced histogram of* u^f*).*

Assuming that $s \in [(1 \pm \epsilon)|\mathcal{S}|]$*,* $\overrightarrow{\mathsf{W1}}(h, h^f) < 4 \cdot \overleftarrow{\mathsf{W1}}(h, h^f) + 100\epsilon$ *and* $\mathsf{W1}(h^{u^f}, h^f) \leq \epsilon$*, then the following hold:*

Completeness: *If* $u = u^f$ *and* $\mathsf{W1}(h^u, h) \leq \epsilon$*, then* $\mathsf{V}_{\mathsf{VEL}}$ *accepts when interacting with* $\mathsf{P}_{\mathsf{VEL}}$ *with probability at least* $1 - 2^{-n/2}$*.*

Soundness: *If* $\mathsf{SH}(u, u^f) > 111\epsilon$*, then (for any unbounded prover)* $\mathsf{V}_{\mathsf{VEL}}$ *rejects with probability at least* $1 - 2^{-n/2}$.

Note about the proofs Throughout our proofs we use the fact that $2^{2\epsilon} > 1 + \epsilon > 2^{\epsilon/2}$ and $2^{-\epsilon/2} > 1 - \epsilon > 2^{-2\epsilon}$ for small $\epsilon > 0$. Namely, for small ϵ enough the range $[1 - \epsilon, 1 + \epsilon]$ is the same as $[2^{-2\epsilon}, 2^{2\epsilon}]$ up to a factor of 2 over ϵ. Therefore we will freely replace expressions of the form $s \in [(1 \pm \epsilon)|\mathcal{S}|]$ with expressions of the form $s \in [2^{\pm\epsilon} \cdot |\mathcal{S}|]$ (and sometimes vice versa) to make the manipulations simpler, with the understanding that this affects the statement of our theorems by introducing an inconsequential loss of constant factor 2 in parameters.

The formal proofs of Lemma 4.4.1 (protocol SampWithSize) and Lemma 4.4.4 (protocol VerHist) are somewhat different than the overview given in Section 4.2. The idea, however, remains the same: for SampWithSize, the prover will claim to us a histogram for D whose correctness we check using image and preimage tests, and then if it is correct we sample many elements plus their sizes from D, check that the empirical histogram of the samples is close to what the prover claimed, and if so output one of them. Instead of using VerHist to perform the image and preimage tests, we will define SampWithSize to explicitly use VerEmpLabel to do the preimage tests and WeakVerHist to do the image tests. (In fact, we will then use SampWithSize to prove VerHist. There will be no circularity since WeakVerHist and VerEmpLabel do not depend on VerHist). It turns out that for the formal proof, this alternative is cleaner to present, and in fact allows us to achieve better parameters, than proving things in the order presented in Section 4.2.

We now move to proving the main lemmas of this section.

Proof of Lemma 4.4.1. We first a give an implementation of SampWithSize and its multi-query case with a **PSPACE** prover. Since they are both constant round

protocols Lemma 4.3.10 yields that there exist an $\mathbf{BPP}^{\mathbf{NP}}$ prover for Protocols SampWithSize and MultSampWithSize with slightly weaker guarantee which are good enough for our purposes to prove Theorem 4.1.1.

Protocol SampWithSize is defined as follows.

Protocol 4.4.7.

$\mathsf{SampWithSize} = (\mathsf{P_{SWS}}, \mathsf{V_{SWS}})$.

Common input: An efficiently decidable set $\mathcal{S} \subseteq \{0,1\}^n$*, accuracy parameter* $\delta = 1/\operatorname{poly}(n)$*, an efficiently computable function* $f : \mathcal{S} \mapsto \{0,1\}^*$ *and a size estimation* s *(for* $|\mathcal{S}|$*).*

Description: In the following let $\epsilon = \delta^2/(1000n)$ *and* $\ell = n/\epsilon^2$*.*

1. ***Sampling random points from*** $\mathcal{S}$***.*** *The parties interact, in an execution of the Uniform Sampling Protocol of Lemma 4.3.14, on input* $(\mathcal{S}^\ell, s^\ell, \delta/9)$*, where* $\mathsf{P_{SWS}}$ *and* $\mathsf{V_{SWS}}$ *play the role of the prover and the verifier respectively. Let* $(x_1, \ldots, x_\ell)$ *denote* $\mathsf{V_{US}}$*'s output in this execution.*

2. ***Compute the histogram and labeling.*** $\mathsf{P_{SWS}}$ *computes* $h \in [0,1]^{m+1}$ *the* ϵ*-histogram of* f *with respect to* S *as well as the labeling* $u\colon [\ell] \mapsto (m)$*, where* $u(i) = \mathsf{Bin}(x_i)$ *(see Definition 4.3.1 for the definition of* $\mathsf{Bin}(x_i)$*).* $\mathsf{P_{SWS}}$ *sends* h, u *to* $\mathsf{V_{SWS}}$*.*

3. ***Verify the claimed histogram (image tests).*** *The parties interact in the* WeakVerHist *protocol (of Lemma 4.4.5) on input* $(\mathcal{S}, s, f, \epsilon, h)$ *where* $\mathsf{P_{SWS}}$ *and* $\mathsf{V_{SWS}}$ *play the role of* $\mathsf{P_{WVH}}$ *and* $\mathsf{V_{WVH}}$ *respectively.*

4. ***Verify the samples (preimage tests).*** $\mathsf{P_{SWS}}, \mathsf{V_{SWS}}$ *engage in the* VerEmpLabel *protocol (of Lemma 4.4.6) on input* $(\mathcal{S}, s, f, \epsilon, h, x_1, \ldots, x_\ell, u)$*. If* VerEmpLabel *rejects, then* $\mathsf{V_{SWS}}$ *rejects as well.*

5. ***Choose output.*** $\mathsf{V}_{\mathsf{SWS}}$ *chooses* $i \xleftarrow{R} [\ell]$ *and outputs* $(x_i, s \cdot 2^{-\epsilon \cdot u(i)})$.

We next prove the completeness and soundness of Protocol 4.4.7. Recall our notation that u^f is the honest labeling of the examples $x_1, \ldots, x_\ell$, $h^{u^f} = \mathsf{Hist}(u^f)$ denotes the empirical histogram given by the honest labeling u^f, $h^u = \mathsf{Hist}(u)$ denotes the empirical histogram given by the claimed labeling u, and h^f denotes the honest histogram of f over $\mathcal{S}$. The intuition of the proof roughly follows in three steps, described below.

1. $x_1, \ldots, x_\ell$ are sampled almost uniformly in $\mathcal{S}$, so we can apply a Chernoff bound to argue that $\mathsf{W1}(h^{u^f}, h^f)$ is small.

2. The $\mathsf{WeakVerHist}$ protocol guarantees that the claimed histogram h satisfies $\overrightarrow{\mathsf{W1}}(h, h^f) < 4\overleftarrow{\mathsf{W1}}(h, h^f) + 100\epsilon$, therefore the $\mathsf{VerEmpLabel}$ protocol guarantees that $\mathsf{SH}(u, u^f)$ is small.

3. We apply the following lemma that states that if $\mathsf{SH}(u, u^f)$ is small, then by picking a random $i \xleftarrow{\mathrm{R}} [\ell]$ we get an output $(x_i, s2^{-\epsilon \cdot u(i)})$ which has the requirements of Lemma 4.4.1.

Lemma 4.4.8. *Suppose that* $\mathsf{SH}(u, u^f) \leq 111\epsilon$*, then*

$$\Pr_{i \xleftarrow{R} [\ell]} \left[s_i \cdot 2^{-\epsilon \cdot u(i)} \in \left[|f^{-1}(f(x_i))| \cdot 2^{\pm\delta} \right] \right] \geq 1 - \delta/9.$$

Proof. The promise that $\mathsf{SH}(u, u^f) \leq 111\epsilon = 111n/m$ together with Proposition 4.3.7 (item 2) yield that $\Pr_{i \xleftarrow{\mathrm{R}} [\ell]}[|u^f(x_i) - u(x_i)| \leq 111n/(\delta/9)] \geq 1 - \delta/9$, and we conclude

that

$$
\begin{aligned}
& s \cdot 2^{-\epsilon \cdot u(i)} \\
& \in 2^{\pm\epsilon} \cdot |\mathcal{S}| \cdot 2^{\epsilon \cdot u^f(x_i)} \cdot 2^{\pm\epsilon \cdot (999n/\delta)} && \text{since } s \in [2^{\pm\gamma} \cdot |\mathcal{S}|] \\
& \subseteq 2^{\pm\epsilon} \cdot 2^{\pm\epsilon} \cdot |f^{-1}(f(x_i))| \cdot 2^{\pm\epsilon \cdot (999n/\delta)} && \text{by Proposition 4.3.3} \\
& = |f^{-1}(f(x_i))| \cdot 2^{\pm\epsilon \cdot (2+999n/\delta)} \\
& \subseteq |f^{-1}(f(x_i))| \cdot 2^{\pm\epsilon \cdot (1000n/\delta)} \\
& = |f^{-1}(f(x_i))| \cdot 2^{\pm\delta}.
\end{aligned}
$$

Thus with probability $1-\delta/9$ we pick a good i satisfying the requirement of Lemma 4.4.1.

□

We now use Lemma 4.4.8 to formally prove that SampWithSize is complete and sound (with an unbounded prover).

Completeness. If $s \in [2^{\pm\gamma} \cdot |\mathcal{S}|]$, then $s^\ell \in [2^{\pm\gamma\ell} \cdot \left|\mathcal{S}^\ell\right|]$. Now note that since $\ell\gamma < (\delta/9)/30$, therefore s satisfies the promise of the Uniform Sampling protocol used in Step 1. By the completeness of Lemma 4.3.14 the verifier rejects in Step 1 with probability at most $\delta/9$.

Moreover, the uniformity condition of Lemma 4.3.14 implies that for all sets $T \subseteq$

$\mathcal{S}^\ell$, it holds that

$$\Pr[(x_1, \ldots, x_\ell) \in T] \tag{4.4.1}$$

$$= (\Pr[(x_1, \ldots, x_\ell) \in T] + \Pr[\mathsf{V}_{\mathsf{SWS}} \text{ rejects }] - \Pr[U_{\mathcal{S}^\ell} \in T]) + \Pr[U_{\mathcal{S}^\ell} \in T] - \Pr[\mathsf{V}_{\mathsf{SWS}} \text{ rejects}]$$

$$= (\Pr[(x_1, \ldots, x_\ell) \in T \cup \{\perp\}] - \Pr[U_{\mathcal{S}^\ell} \in T \cup \{\perp\}]) + \Pr[U_{\mathcal{S}^\ell} \in T] - \Pr[\mathsf{V}_{\mathsf{SWS}} \text{ rejects}] \tag{4.4.2}$$

$$\leq \Delta((x_1, \ldots, x_\ell), U_{\mathcal{S}^\ell}) + \Pr[U_{\mathcal{S}^\ell} \in T] - \Pr[\mathsf{V}_{\mathsf{SWS}} \text{ rejects}] \tag{4.4.3}$$

$$\leq \Pr[U_{\mathcal{S}^\ell} \in T] + \delta/9 \tag{4.4.4}$$

In the above, the probability is over the $(x_1, \ldots, x_\ell)$ sampled by $\mathsf{V}_{\mathsf{SWS}}$ as in Step 1, which can possibly be $\perp$ if $\mathsf{V}_{\mathsf{SWS}}$ rejects. In Inequality 4.4.2 we use the fact that samples from $U_{\mathcal{S}}$ never equal $\perp$. Inequality 4.4.3 follows by the definition of statistical distance. Inequality 4.4.4 follows by the uniformity condition of Lemma 4.3.14.

The following claim asserts that the empirical histogram of uniform samples from $U_{\mathcal{S}^\ell}$ are close to the true histogram with high probability.

Claim 4.4.9. *Let $T \subseteq \mathcal{S}^\ell$ be the set of tuples $(x_1, \ldots, x_\ell)$ such that the true empirical histogram h^{u^f} satisfies $\mathsf{W1}(h^{u^f}, h^f) > \epsilon$. Then*

$$\Pr[U_{\mathcal{S}^\ell} \in T] < m2^{-n}$$

Proof. It follows that for each $j \in (m-1)$ that

$$\Pr_{x \xleftarrow{\mathrm{R}} \mathcal{S}}[\mathsf{Bin}(x) \leq j] = \sum_{i \in (j)} h_i^f$$

Let X_i^j be the random variable such that $X_i^j = 1$ if $\mathsf{Bin}(x_i) \leq j$, and $X_i^j = 0$ otherwise.

Since $\sum_{i\in(j)} h_i^{u^f} = \frac{1}{\ell} \cdot \sum_{i\in[\ell]} X_i^j$, applying a Chernoff bound yields that

$$\begin{aligned}
&\Pr\left[\left|\Big(\sum_{i\in(j)} h_i^{u^f}\Big) - \Big(\sum_{i\in(j)} h_i^f\Big)\right| > \epsilon\right] \\
&= \Pr\left[\left|\tfrac{1}{\ell}\sum_{i=1}^{\ell} X_i^j - \Big(\sum_{i\in(j)} h_i^f\Big)\right| > \epsilon\right] \\
&\leq 2e^{-\ell\epsilon^2} = 2e^{-n} < 2^{-n}.
\end{aligned}$$

It follows that

$$\begin{aligned}
&\Pr[U_{\mathcal{S}^\ell} \in T] \\
&= \Pr[\mathsf{W1}\Big(h^{u^f}, h^f\Big) > \epsilon] = \Pr\left[\sum_{j\in(m)} \left|\Big(\sum_{i\in(j)} h_i^{u^f}\Big) - \Big(\sum_{i\in(j)} h_i^f\Big)\right| > m\epsilon\right] \\
&\leq \left(\sum_{j\in(m-1)} \Pr\left[\left|\Big(\sum_{i\in(j)} h_i^{u^f}\Big) - \Big(\sum_{i\in(j)} h_i^f\Big)\right| > \epsilon\right]\right) \leq m2^{-n}.
\end{aligned}$$

□

Since the prover is honest, $h = h^f$ and therefore it follows by the completeness of $\mathsf{WeakVerHist}$ that $\mathsf{V_{SWS}}$ rejects in Step 3 the with probability at most $2^{-n/2}$.

Since the prover is honest, $h = h^f$ and $u = u^f$. Therefore by Claim 4.4.9 and Inequality 4.4.4, it holds that $\Pr[(x_1, \ldots, x_\ell) \in T] \leq m2^{-n} + \delta/9$. Therefore, suppose $(x_1, \ldots, x_\ell) \notin T$, which implies that $\mathsf{W1}(h^{u^f}, h^f) \leq \epsilon$. Along with the promise that $s \in [2^{\pm\gamma} \cdot |\mathcal{S}|] \subseteq [2^{\pm\epsilon} \cdot |\mathcal{S}|]$ holds, and the fact that $h = h^f$, we can apply the completeness of Lemma 4.4.6 to deduce that the verifier rejects in Step 4 with probability at most $2^{-n/2}$.

So if the prover is honest the verifier does not reject in any of the steps with probability $\leq 2\delta/9 + 2 \cdot 2^{-n/2} < \delta$.

Soundness. Fix a cheating prover P^*. Let $\mathsf{BadSize}$ be the bad event that $s_i 2^{-\epsilon u(i)} \notin [2^{\pm\delta}|f^{-1}(f(x_i))|]$ and let $\mathsf{NoReject}$ be the event that $\mathsf{V_{SWS}}$ does not reject. Let $\mathsf{NoReject}_i$ be the event that $\mathsf{V_{SWS}}$ does not reject in Steps $1, \ldots, i$ (but may possibly reject in later steps), and finally let $\mathsf{BadHist}$ be the event that $(x_1, \ldots, x_\ell) \in T$ as defined in Claim 4.4.9. Our goal is to bound $\Pr[\mathsf{BadSize} \wedge \mathsf{NoReject}]$. It holds that

$$\begin{aligned}
&\Pr[\mathsf{BadSize} \wedge \mathsf{NoReject}] && (4.4.5)\\
&\leq \Pr[\mathsf{BadSize} \wedge \mathsf{NoReject}_1]\\
&= \Pr[\mathsf{BadSize} \wedge \mathsf{NoReject}_1 \wedge \mathsf{BadHist}] + \Pr[\mathsf{BadSize} \wedge \mathsf{NoReject}_1 \wedge \overline{\mathsf{BadHist}}]\\
&\leq \Pr[\mathsf{BadHist}] + \Pr[\mathsf{BadSize} \mid \mathsf{NoReject}_1 \wedge \overline{\mathsf{BadHist}}] && (4.4.6)
\end{aligned}$$

As in the analysis of completeness, Claim 4.4.9 and Inequality 4.4.4 yield that $\Pr[\mathsf{BadHist}] \leq m2^{-n} + \delta/9$. In order to bound the second probability in Inequality 4.4.6, it suffices to bound the sum of the probabilities of the following events conditioned on $\mathsf{NoReject}_1 \wedge \overline{\mathsf{BadHist}}$:

1. The probability that $\overrightarrow{\mathsf{W1}}(h^f, h) > 4\overleftarrow{\mathsf{W1}}(h^f, h) + 100\epsilon$ and $\mathsf{WeakVerHist}$ accepts.
2. Conditioned on all previous events not occurring, the probability that $\mathsf{SH}(u, u^f) > 111\epsilon$ and $\mathsf{VerEmpLabel}$ accepts.
3. Conditioned on all previous events not occurring, the probability that the output $s_i 2^{-\epsilon u(i)} \notin [2^{\pm\delta}|f^{-1}(f(x_i))|]$ (i.e., $\mathsf{BadSize}$ occurs).

By conditioning on $\mathsf{NoReject}_1 \wedge \overline{\mathsf{BadHist}}$, all the promises required by Lemma 4.4.5 are satisfied, therefore by the soundness of Lemma 4.4.5 the first item occurs with probability $< 2^{-n/2}$. Then, in the second item the promises of Lemma 4.4.6 are satisfied, so the soundness of Lemma 4.4.6 implies that this event occurs with probability $< 2^{-n/2}$. Finally, the Correctness Lemma 4.4.8 implies that the last event

occurs with probability $\delta/9$. Therefore the conditional probability in Inequality 4.4.6 is bounded by $2^{-n/2+1}+\delta/9$, and the entire probability in Inequality 4.4.6 is bounded by $m2^{-n}+2\delta/9+2^{-n/2+1}<\delta$.

For the second part of the soundness, let $\mathsf{Reject}=\overline{\mathsf{NoReject}}$ and $\mathsf{Reject}_i=\overline{\mathsf{NoReject}}_i$. By the uniformity guarantee of Lemma 4.3.14, it holds for every i, that

$$\Delta(x_i,U_{\mathcal{S}})\leq \Pr[\mathsf{Reject}_1]+\delta/9$$

and it particular it holds if i is chosen at random. Note that $\mathsf{Reject}_i\cap\mathsf{Reject}_j=\varnothing$ for any $i\neq j$. Therefore if as a mental experiment we choose $i\xleftarrow{R}[\ell]$ at the beginning (rather than the last step), the final output x satisfies:

$$\begin{aligned}\Delta(x,U_{\mathcal{S}}) &= \Delta(x_i,U_{\mathcal{S}})+\Pr[\mathsf{Reject}_2\vee\mathsf{Reject}_3\vee\mathsf{Reject}_4]\\ &\leq \Pr[\mathsf{Reject}_1]+\delta/9+\Pr[\mathsf{Reject}_2\vee\mathsf{Reject}_3\vee\mathsf{Reject}_4]\\ &= \delta/9+\Pr[\mathsf{Reject}]\\ &= p+\delta/9,\end{aligned}$$

which concludes the proof. □

Next, we prove the multiple-query version of the above lemma.

Proof. (of Lemma 4.4.2) Protocol $\mathsf{MultSampWithSize}$ is defined as follows.

Protocol 4.4.10.

$\mathsf{MultSampWithSize}=(\mathsf{P}_{\mathsf{MSWS}},\mathsf{V}_{\mathsf{MSWS}})$.

Common input: An accuracy parameter δ, and for every $i\in[k]$: an efficiently decidable set $\mathcal{S}_i$, efficiently computable function $f_i\colon\mathcal{S}_i\mapsto\{0,1\}^*$ and a size estimation s_i.

Description: In the following let $\mathcal{S} = \mathcal{S}_1 \times \cdots \times \mathcal{S}_k \subseteq \{0,1\}^n$*, let* $g(x_1, \ldots, x_k) := (f_1(x_1), \ldots, f_k(x_k))$ *and let* $s = s_1 \cdots s_k$*.*

1. ***Sampling points jointly.*** *The parties interact in an execution of the* SampWithSize *protocol of Lemma 4.4.1 on input* $(\mathcal{S}, s, g, \delta/2)$*, where* $\mathsf{P}_{\mathsf{SWS}}$ *and* $\mathsf{V}_{\mathsf{SWS}}$ *play the role of the prover and the verifier respectively. Let* $(x = (x_1, \ldots, x_k), s_x)$ *denote* $\mathsf{V}_{\mathsf{SWS}}$*'s output.*

2. ***Sending the sibling sizes.*** $\mathsf{P}_{\mathsf{MSWS}}$ *sends* $s_{x_i} = |f_i^{-1}(f_i(x_i))|$ *for every* $i \in [k]$ *to* $\mathsf{V}_{\mathsf{MSWS}}$*, and* $\mathsf{V}_{\mathsf{MSWS}}$ *rejects if* $s_x \neq s_{x_1} \cdots s_{x_k}$*.*

3. ***Lower-bound test.*** *For every* $i \in [k]$ *in parallel, the parties interact in an execution of the lower-bound protocol of Lemma 4.3.12 on input* $(f_i^{-1}(f_i(x_i)), s_{x_i}, \delta/8k)$*, where* $\mathsf{P}_{\mathsf{SWS}}$ *and* $\mathsf{V}_{\mathsf{SWS}}$ *play the role of the prover and the verifier respectively.*

4. ***Output.*** $\mathsf{V}_{\mathsf{SWS}}$ *outputs* $((x_1, s_{x_1}), \ldots, (x_k, s_{x_k}))$*.*

Since $s_i \in [(1 \pm \gamma) \cdot |\mathcal{S}_i|] \subseteq [2^{\pm 2\gamma} \cdot |\mathcal{S}_i|]$ for every $i \in [k]$, it follows that $s \in [2^{\pm 2k\gamma} \cdot |\mathcal{S}|] \subseteq [(1 \pm 8k\gamma) \cdot |\mathcal{S}|]$. Therefore, s satisfies the promise of Lemma 4.4.1 for accuracy parameter $\delta/2$ (recall that $\gamma := \frac{1}{8k} \cdot (\frac{\delta}{20n})^8$).

Completeness. When interacting with the honest prover, Lemma 4.4.1 yields that $\mathsf{V}_{\mathsf{MSWS}}$ rejects in Step 1 with probability at most $\delta/2$. Clearly, $\mathsf{V}_{\mathsf{MSWS}}$ does not reject in Step 2, and Lemma 4.3.12 and a union bound yield that $\mathsf{V}_{\mathsf{MSWS}}$ rejects in Step 3 with probability at most $k \cdot \mathrm{neg}(n) = \mathrm{neg}(n)$. Therefore, the probability that $\mathsf{V}_{\mathsf{MSWS}}$ rejects is bounded by $\delta/2 + \mathrm{neg}(n) < \delta$.

Soundness. In the following we assume without loss of generality that $s_x = s_{x_1} \cdots s_{x_k}$ (as otherwise $\mathsf{V_{SWS}}$ rejects). Lemma 4.4.1 yields that

$$\Pr[\overline{E_1} \wedge \mathsf{V_{SWS}} \text{ does not reject at Step 1}] < \delta/2, \tag{4.4.7}$$

where E_1 is the event that $s_x \in (1 \pm \delta/2) \cdot |g^{-1}(g(x))|$, and Lemma 4.3.12 yields that

$$\Pr[\overline{E_2} \wedge \mathsf{V_{SWS}} \text{ does not reject at Step 3}] \leq \mathrm{neg}(n), \tag{4.4.8}$$

where E_2 is the event that $s_{x_i} \leq (1 + \delta/8k) \cdot |f_i^{-1}(f_i(x_i))|$ for every $i \in [k]$.

When E_2 occurs, then for every $i \in [k]$ it holds (using the fact that $(1+\delta/(8k))^{k-1} \leq 1 + \delta/2$) that

$$\Pi_{j \neq i} s_{x_j} \leq (1 + \delta/2) \cdot \Pi_{j \neq i} |f_j^{-1}(f_j(x_j))|$$

Since $|g^{-1}(g(x))| = \Pi_i |f_i^{-1}(f_i(x_i))|$, therefore the following holds for all $i \in [k]$ when $E_1 \wedge E_2$ occurs:

$$s_{x_i} = \frac{s_x}{\prod_{j \neq i} s_{x_i}} \geq \frac{(1 - \delta/2) \cdot \prod_{j=1}^{k} |f_j^{-1}(f_j(x_j))|}{(1 + \delta/2) \prod_{j \neq i} |f_j^{-1}(f_j(x_j))|} \geq (1 - \delta)|f_i^{-1}(f_i(x_i))| \tag{4.4.9}$$

Hence it follows that:

$$\begin{aligned}
&\Pr[\exists i \colon s_{x_i} \cdot (1 \pm \delta) \cdot |f_i^{-1}(f_i(x_i))| \wedge \mathsf{V_{SWS}} \text{ does not reject}] \\
&\leq \Pr[\mathsf{V_{SWS}} \text{ does not reject} \wedge \overline{E_1 \wedge E_2}] \\
&\leq \delta/2 + \mathrm{neg}(n) < \delta.
\end{aligned}$$

For the second part of the soundness proof, Lemma 4.4.1 yields that

$$\mathsf{SD}((U_{\mathcal{S}_1}, \ldots, U_{\mathcal{S}_k}), (x_1, \ldots, x_k)) \leq \delta/2 + \Pr[\mathsf{V_{MSWS}} \text{ rejects in Step 1}].$$

As in the proof ofLemma 4.4.1, since the event that $\mathsf{V}_{\mathsf{MSWS}}$ rejects in later steps is disjoint from the event that $\mathsf{V}_{\mathsf{MSWS}}$ rejects in Step 1, it follows that

$$\mathsf{SD}((U_{\mathcal{S}_1},\ldots,U_{\mathcal{S}_k}),(x_1,\ldots,x_k)) \leq \delta/2 + \Pr[\mathsf{V}_{\mathsf{MSWS}} \text{ rejects}]$$

□

Proof. (of Lemma 4.4.4) The protocol VerHist described below only achieves completeness and soundness $O(\epsilon)$. This this can be easily amplified, however, to error below 2^{-n} via parallel repetition.

Protocol 4.4.11. $\mathsf{VerHist} = (\mathsf{P}_{\mathsf{VH}}, \mathsf{V}_{\mathsf{VH}})$.

Common input: An efficiently decidable set $\mathcal{S} \subseteq \{0,1\}^n$, an efficiently computable function $f : \mathcal{S} \mapsto \{0,1\}^*$, a size estimation s (for $|\mathcal{S}|$), the histogram parameter $0 < \epsilon < 1$ and a (claimed) ϵ-histogram h of f with respect to $\mathcal{S}$.

Description: Note that it holds that $m = n/\epsilon$ where $h \in [0,1]^{m+1}$. In the following let $\ell = 100m^2n = 100n^3/\epsilon^2$.

1. ***Sample ℓ random elements from $\mathcal{S}$.*** *The parties interact in an execution of the parallel* SampWithSize *protocol of Lemma 4.4.2, on input $(\epsilon,(\mathcal{S}_1,s,f),\ldots,(\mathcal{S}_\ell,s,f))$, where $\mathcal{S}_i = \mathcal{S}$ for all $i \in [\ell]$ and P_{VH} and V_{VH} play the role of $\mathsf{P}_{\mathsf{MSWS}}$ and $\mathsf{V}_{\mathsf{MSWS}}$ respectively. Let $((x_1,s_{x_1}),\ldots,(x_\ell,s_{x_\ell}))$ be the result of the interaction.*

2. ***Approximate the histogram.*** *For $i \in [\ell]$, let $u(i) = \lfloor(\log(s_{x_i}/s))/\epsilon\rfloor$. Let $h^u = \mathsf{Hist}(u)$ be the empirical histogram concluded from the mapping u according to Definition 4.3.2.*

3. ***Verify the claim.*** *Reject if $\mathsf{W1}(h,h^u) \geq 10/m$, and accept otherwise.*

Let $h = h^f$, let $u^f(i) = \mathsf{Bin}(x_i)$ indicate the real bin number of x_i according to Definition 4.3.1, and let h^{u^f} be the empirical histogram concluded from u^f.

Notice that $\gamma \ll \frac{1}{8\ell} \cdot (\frac{\delta}{20n})^8$ and so s is in the right range to be used for MultSampWithSize protocol in Step 1.

We will show that for any prover strategy with high probability either the verifier rejects or h^u is a "good" approximation for h^f in the 1st Wasserstein distance. Then comparing h^u to h to decide accept or reject (Step 3) works properly.

Let $T_1 \subset (\mathcal{S} \times \mathbb{N})^\ell$ be the "bad set" of tuples $((x_1, s_{x_1}), \ldots, (x_\ell, s_{x_\ell}))$ where there exist $i \in [\ell]$ such that $s_{x_i} \notin [2^{\pm\epsilon}|f^{-1}(f(x_i))|]$.

In the following we assume without loss of generality that the x_i's are elements of $\mathcal{S}$.

Claim 4.4.12. *If* $((x_1, s_{x_1}), \ldots, (x_\ell, s_{x_\ell})) \notin T_1$ *then* $\mathsf{W1}(h^u, h^{u^f}) \leq 6/m$.

Proof. If $((x_1, s_{x_1}), \ldots, (x_\ell, s_{x_\ell})) \notin T_1$ then for all $i \in [\ell]$ it holds that $s_{x_i} \in [2^{\pm\epsilon}|f^{-1}(f(x_i))|]$. Therefore

$$\begin{aligned} u(i) &= \lfloor (\log(s_{x_i}/s))/\epsilon \rfloor \\ &\in [(\log(s_{x_i}/s))/\epsilon \pm 1] \\ &\subseteq [(\log(2^{\pm\epsilon}|f^{-1}(f(x_i))|/s))/\epsilon \pm 1] \\ &= [\pm 1 + (\log(|f^{-1}(f(x_i))|/s))/\epsilon \pm 1] \\ &\subseteq \{\mathsf{Bin}(x_i) \pm 3\}. \end{aligned}$$

But if $u(i) \in \{\mathsf{Bin}(x_i) \pm 3\}$ for all $i \in [\ell]$, then by Proposition 4.3.7 and Lemma 4.3.8 it holds that

$$\mathsf{W1}(h^u, h^{u^f}) \leq \mathsf{SH}(u, u^f) = \overleftarrow{\mathsf{SH}}(u, u^f) + \overrightarrow{\mathsf{SH}}(u, u^f) \leq 3/m + 3/m = 6/m. \quad (4.4.10)$$

□

Let $T_2 \subseteq \mathcal{S}^\ell$ be another "bad set", the set of tuples $(x_1, \ldots, x_\ell)$ such that $\mathsf{W1}(h^{u^f}, h^f) > 1/m$. The following claim can be proven similar to Claim 4.4.9:

Claim 4.4.13. $\Pr[U_{\mathcal{S}^\ell} \in T_2] < m2^{-n} = \mathrm{neg}(n)$.

In the following let $\mathsf{NoReject}_1$ be the event that V_{VH} does not reject in Step 1, let Reject_1 be the event that V_{VH} rejects in Step 1, and let Reject be the event that V_{VH} rejects in some step.

Claim 4.4.14. *Let $((x_1, s_{x_1}), \ldots, (x_\ell, s_{x_\ell}))$ be the result of Step 1. Then it holds that*

$$\Pr[\mathsf{NoReject}_1 \wedge \mathsf{W1}(h^u, h^f) \geq 10/m] \leq 3\epsilon$$

Proof. Note that if $((x_1, s_{x_1}), \ldots, (x_\ell, s_{x_\ell})) \notin T_1$ and $(x_1, \ldots, x_\ell) \notin T_2$ then by the definition of T_2 and Claim 4.4.12 it holds that

$$\mathsf{W1}(h^u, h^f) \leq \mathsf{W1}(h^u, h^{u^f}) + \mathsf{W1}(h^{u^f}, h^f) \leq 1/m + 6/m < 10/m.$$

So it holds that:

$$\begin{aligned}
&\Pr[\mathsf{NoReject}_1 \wedge \mathsf{W1}(h^u, h^f) \geq 10/m] \\
&\leq \Pr[\mathsf{NoReject}_1 \wedge (((x_1, s_{x_1}), \ldots, (x_\ell, s_{x_\ell})) \in T_1 \vee (x_1, \ldots, x_\ell) \in T_2)] \\
&\leq \Pr[\mathsf{NoReject}_1 \wedge ((x_1, s_{x_1}), \ldots, (x_\ell, s_{x_\ell})) \in T_1] + \Pr[\mathsf{NoReject}_1 \wedge (x_1, \ldots, x_\ell) \in T_2].
\end{aligned}$$

Now Claim 4.4.14 will follow by proving the following inequalities.

$$\Pr[\mathsf{NoReject}_1 \wedge ((x_1, s_{x_1}), \ldots, (x_\ell, s_{x_\ell})) \in T_1] \leq \epsilon \tag{4.4.11}$$

$$\Pr[\mathsf{NoReject}_1 \wedge (x_1, \ldots, x_\ell) \in T_2] \leq \epsilon + \mathrm{neg}(n) \tag{4.4.12}$$

The soundness of Lemma 4.4.2 yields that

$$\Pr[\mathsf{V}_{\mathsf{MSWS}} \text{ does not reject} \wedge \exists i \in [\ell] \colon s_{x_i} \notin [2^{\pm\epsilon}|f^{-1}(f(x_i))|] \leq \epsilon$$

which is equivalent to Inequality 4.4.11.

The soundness of Lemma 4.4.2 also yields that

$$\mathsf{SD}(U_{\mathcal{S}^\ell}, (x_1, \ldots, x_\ell)) \leq \epsilon + \Pr[\mathsf{V}_{\mathsf{MSWS}} \text{ rejects}]. \tag{4.4.13}$$

By using Lemma 1.3.4 over Inequality 4.4.13 with parameters $Y = U_{\mathcal{S}^\ell}, X = (x_1, \ldots, x_\ell), T = T_2$ and $\delta = \epsilon$ we get that

$$\Pr[\mathsf{NoReject}_1 \wedge (x_1, \ldots, x_\ell) \in T_2] \leq \epsilon + \Pr[U_{\mathcal{S}^\ell} \in T_2].$$

But Claim 4.4.13 yields that $\Pr[U_{\mathcal{S}^\ell} \in T_2] \leq \mathrm{neg}(n)$ and so Inequality 4.4.12 follows.

□

Now we will prove that Protocol VerHist is complete and sound as described in Lemma 4.4.4 (we prove this for completeness/soundness error $O(\epsilon)$, but this can be amplified to 2^{-n} by repeating in parallel).

Completeness. Let the prover be honest and $h = h^f$. By Claim 4.4.14 it holds that

$$\Pr[\mathsf{Reject}_1 \vee \mathsf{W1}(h^u, h^f) \leq 10/m] \geq 1 - 3\epsilon.$$

By the completeness of Lemma 4.4.2, V_{VH} rejects in Step 1 with probability at most $\delta = \epsilon$ and so

$$\Pr[\mathsf{W1}(h^u, h^f) \leq 10/m] \geq 1 - 4\epsilon.$$

But if $\mathsf{W1}(h^u, h^f) \leq 10/m$, then V_{VH} does not reject in Step 3. Therefore by a union bound:

$$\Pr[\mathsf{Reject}] \leq \epsilon + 4\epsilon = 5\epsilon.$$

Soundness. Suppose $\mathsf{W1}(h^f, h) > 20/m$.

By Claim 4.4.14 it holds that

$$\Pr[\mathsf{Reject}_1 \vee \mathsf{W1}(h^u, h^f) \leq 10/m] \geq 1 - 3\epsilon.$$

But if $\mathsf{W1}(h^f, h^u) < 10/m$, then it would hold that

$$\mathsf{W1}(h^u, h) > \mathsf{W1}(h^f, h) - \mathsf{W1}(h^f, h^u) > 20/m - 10/m = 10/m.$$

So it holds that

$$\Pr[\mathsf{Reject}_1 \vee \mathsf{W1}(h^u, h) > 10/m] \geq 1 - 3\epsilon.$$

Finally since $\mathsf{W1}(h^u, h) > 10/m$ makes the verifier reject in Step 3 therefore:

$$\Pr[\mathsf{Reject}] \geq 1 - 3\epsilon.$$

□

4.4.1 Proving Supporting Lemmas

In this section we prove Lemma 4.4.5 and Lemma 4.4.6.

Proof of Lemma 4.4.5

Proof. We will use the following definition:

Definition 4.4.15 (Exponential sum)**.** Given $x \in \mathbb{R}^{j+1}$ and $\epsilon > 0$, we define the exponential sum of x as $\mathsf{ES}_\epsilon(x) = \sum_{i\in(j)} x_i \cdot 2^{i\epsilon}$ and the normalized exponential sum of x as $\widetilde{\mathsf{ES}}_\epsilon(x) = \mathsf{ES}_\epsilon(x)/2^{j\epsilon}$. When ϵ is clear from the context, we omit ϵ from the notation.

Let h^f be the ϵ-histogram of a function f over $\{0,1\}^n$, Proposition 4.3.3 yields that $\mathsf{ES}_\epsilon(h^f{}_{\leq j}) \leq |\bigcup_{i\in(j)} \mathcal{B}_i| \leq 2^\epsilon \mathsf{ES}_\epsilon(h^f{}_{\leq j})$.

Protocol 4.4.16. $\mathsf{WeakVerHist} = (\mathsf{P}_{\mathsf{WVH}}, \mathsf{V}_{\mathsf{WVH}})$.

Common input: An efficiently decidable set $\mathcal{S} \subseteq \{0,1\}^n$, a size estimation s (for $|\mathcal{S}|$), a function $f : \mathcal{S} \mapsto \{0,1\}^*$, a histogram parameter $\epsilon = 1/\operatorname{poly}(n)$, and h a (claimed) ϵ-histogram of f over $\mathcal{S}$.

Description: For simplicity we assume that $f(\mathcal{S}) \subseteq \{0,1\}^n$.[9] *For each $j \in (m)$, let M^j be the following protocol: on input $y \in \{0,1\}^n$, if $y \notin f(\mathcal{S})$, the verifier rejects. Otherwise, the parties engage in the Set-Lower-Bound protocol of Lemma 4.3.12 with input $(f^{-1}(y), s \cdot 2^{-\epsilon\cdot(j+2)}, \epsilon)$. Protocol* WeakVerHist *is defined as follows:*

1. *The parties interact for every $j \in (m)$, in parallel, in an execution of the of Lemma 4.3.13, on input $(1^n, \mathsf{M}^j, \mathsf{ES}_\epsilon(h_{\leq j}), \epsilon)$, where $\mathsf{V}_{\mathsf{WVH}}$ and $\mathsf{P}_{\mathsf{WVH}}$ play the role of $\mathsf{V}_{\mathsf{GLB}}$ and $\mathsf{P}_{\mathsf{GLB}}$ respectively.*

[9] Since f is efficiently computable, it holds that $f(\mathcal{S}) \subseteq \{0,1\}^{\operatorname{poly}(n)}$ and all the proof can easily be adapted to this case as well.

2. $\mathsf{V}_{\mathsf{WVH}}$ *accepts if* $\mathsf{V}_{\mathsf{GLB}}$ *accepts in all the above embedded executions.*

. . . . . .

We next prove the completeness and soundness of Protocol 4.4.16. In the following we fix $\mathcal{S}$, f, n, s and ϵ, let $\mathcal{B}_j$'s be as in Definition 4.3.1 with respect to to this fixing. Also let $(\mathcal{Y}^j, \mathcal{N}^j, \mathcal{T}^j)$ be, in order, the YES, NO, and non-promise inputs of length n for the protocol M^j according to Definition 1.3.6 (since n is fixed, we will not write the index n). Finally, let $\mathsf{GenSetLB}^j$ be the j'th execution of the Generalized Set-Lower-bound protocol done in a random execution of $\mathsf{WeakVerHist}$.

Claim 4.4.17. *It holds that:*

1. $\bigcup_{i\in(j)} \mathcal{B}_i \subseteq \mathcal{Y}^j$, *and*
2. $\mathcal{Y}^j \cup \mathcal{T}^j \subseteq \bigcup_{1\le i\le j+3} \mathcal{B}_i$.

Proof. Let $y \in \mathcal{B}_i$. First consider the case that $i \le j$. Proposition 4.3.3 yields that $|\mathcal{S}| \cdot 2^{-\epsilon\cdot(j+1)} \le |\mathcal{S}| \cdot 2^{-\epsilon\cdot(i+1)} < |f^{-1}(y)|$, and by the promise on s it holds that $s \cdot 2^{-\epsilon\cdot(j+2)} \le |f^{-1}(y)|$. The completeness of Lemma 4.3.12 yields that verifier accepts in M^j with probability at least $1 - 2^{-n}$ and therefore $\mathcal{B}_i \subseteq \mathcal{Y}^j$.

Now let $i \ge j+4$. Proposition 4.3.3 yields that $|f^{-1}(y)| \le |\mathcal{S}|\cdot 2^{-\epsilon i} \le |\mathcal{S}|\cdot 2^{-\epsilon\cdot(j+4)}$. Since $|\mathcal{S}| \le s \cdot 2^{\epsilon}$, it holds that $|f^{-1}(y)| \le 2^{-\epsilon}(s \cdot 2^{-\epsilon\cdot(j+2)})$. Therefore, the soundness of Lemma 4.3.12 yields that the verifier rejects in M^j with probability at least $1-2^{-n}$, which implies that $\mathcal{B}_i \subseteq \mathcal{N}^j$. Also note that by the definition of M^j, it holds that $\{0,1\}^n \setminus f(\mathcal{S}) \subseteq \mathcal{N}^j$, and therefore

$$\mathcal{Y}^j \cup \mathcal{T}^j \subseteq f(\mathcal{S}) \setminus \bigcup_{j+4\le i\le m} \mathcal{B}_i = \bigcup_{1\le i\le j+3} \mathcal{B}_i.$$

□

Completeness. Assuming that $h = h^f$, Proposition 4.3.3 and Claim 4.4.17 yield that $\mathsf{ES}(h_{\leq j}) = \mathsf{ES}(h^f_{\leq j}) \leq |\bigcup_{i\in(j)} \mathcal{B}_i| \leq |\mathcal{Y}^j|$. Therefore, the completeness of Lemma 4.3.13 yields that the verifier accepts in $\mathsf{GenSetLB}^j$ with probability at least $1 - 2^{-n}$, and by the union bound $\mathsf{V}_{\mathsf{WVH}}$ accepts in all of the $m+1$ instances of $\mathsf{GenSetLB}^j$'s (simultaneously) with probability at least $1 - (m+1)2^{-n} > 1 - 2^{-n/2}$.

Soundness. The following lemma carries the heart of the proof.

Lemma 4.4.18. *Let d and d' be two probability distributions over $[0,1]^{m+1}$ and let $\epsilon, \lambda \in (0,1)$. Assume that*

1. *$m \cdot \epsilon \geq 1$ and*
2. *$\mathsf{ES}_\epsilon(d'_{\leq j}) \leq 2^\lambda \cdot \mathsf{ES}_\epsilon(d_{\leq j})$ for all $j \in (m)$,*

then $\overrightarrow{\mathsf{W1}}(d, d') \leq 16\lambda + 4\overleftarrow{\mathsf{W1}}(d, d')$.

Before proving Lemma 4.4.18, we first use it to show the soundness of Protocol 4.4.16. Let $m' = m + 3$, $d' = (d'_0, \ldots, d'_{m'}) = (0, 0, 0, h)$ and $d = (d_0, \ldots, d_{m'}) = (h^f, 0, 0, 0)$ (notice that we are using Lemma 4.4.18 with dimension $m'+1$ rather than $m+1$). Since $m'\epsilon > m\epsilon = n \geq 1$, the first condition of Lemma 4.4.18 is satisfied. The following claim yields that (d, d') also satisfies the second condition of Lemma 4.4.18 for the suitable choice of λ.

Claim 4.4.19. *For any $j \in (m)$ and any prover P^*, either $\mathsf{ES}(h_{\leq j}) \leq 2^{2\epsilon} \cdot \mathsf{ES}(h^f_{\leq j+3})$ or $\mathsf{V}_{\mathsf{WVH}}$ rejects in $\mathsf{GenSetLB}^j$ with probability at least $1 - 2^{-n}$ (where we let $h^f_i = 0$ for $i > m$).*

Proof. Assuming that $\mathsf{ES}(h^f_{\leq j+3}) < 2^{-2\epsilon} \cdot \mathsf{ES}(h_{\leq j})$, Claim 4.4.17 together with Proposition 4.3.3 yield that

$$|\mathcal{Y}^j \cup \mathcal{T}^j| \leq \left| \bigcup_{i \in (j+3)} \mathcal{B}_i \right| < 2^{\epsilon} \cdot \mathsf{ES}(h^f_{\leq j+3}) < 2^{\epsilon} 2^{-2\epsilon} \cdot \mathsf{ES}(h_{\leq j}) = 2^{-\epsilon} \mathsf{ES}(h_{\leq j}).$$

Therefore, Lemma 4.3.13 yields that $\mathsf{V}_{\mathsf{WVH}}$ rejects in $\mathsf{GenSetLB}^j$ with probability at least $1 - 2^{-n}$. □

For $j \in \{0, 1, 2\}$ and any $\lambda > 0$, it holds that $\mathsf{ES}(d'_{\leq j}) = 0 \leq 2^{\lambda} \cdot \mathsf{ES}(d_{\leq j})$. Claim 4.4.19 yields that either $\mathsf{V}_{\mathsf{WVH}}$ rejects with probability at least $1 - 2^{-n}$ or it holds that

$$\mathsf{ES}(d'_{\leq j}) = 0 + 2^{3\epsilon} \cdot \mathsf{ES}(h_{\leq j-3}) \leq 2^{3\epsilon} 2^{2\epsilon} \mathsf{ES}(h^f_{\leq j}) = 2^{5\epsilon} \mathsf{ES}(d_{\leq j}),$$

for $3 \leq j \leq m'$. Hence, the second requirement of Lemma 4.4.18 holds for $\lambda = 5\epsilon$. Below, by $\overrightarrow{\mathsf{W1}}_{m'}(h^f, h)$ we mean the 1st Wasserstein distance when the dimension is increased m' (by adding three zeros in coordinates $m+1, m+2, m+3$). Note $m' \cdot \mathsf{W1}_{m'}(h^f, h) = m \cdot \mathsf{W1}(h^f, h)$ and that $\mathsf{W1}_{m'}(d', h) \leq 3/m < 3\epsilon$. We conclude that

$$
\begin{aligned}
&\overrightarrow{\mathsf{W1}}(h^f, h) \\
&= (m'/m)\overrightarrow{\mathsf{W1}}_{m'}(h^f, h) && \text{change } m \to m' \\
&\leq (m'/m)(\overrightarrow{\mathsf{W1}}(h^f, d) + \overrightarrow{\mathsf{W1}}(d, d') + \overrightarrow{\mathsf{W1}}(d', h)) && \text{by Proposition 4.3.5} \\
&\leq (m'/m)(0 + 16\lambda + 4\overleftarrow{\mathsf{W1}}(d, d') + 3\epsilon) && \text{by Lemma 4.4.18} \\
&\leq (m'/m)(16(5\epsilon) + 4(\overleftarrow{\mathsf{W1}}(d, h^f) + \overleftarrow{\mathsf{W1}}_{m'}(h^f, h) + \overleftarrow{\mathsf{W1}}(h, d')) + 3\epsilon) && \text{by Proposition 4.3.5} \\
&= (m'/m) \cdot (83\epsilon) + 4\overleftarrow{\mathsf{W1}}(h^f, h) && \text{change } m' \to m \\
&\leq 100\epsilon + 4\overleftarrow{\mathsf{W1}}(h^f, h). && \text{for } m \geq n > 15
\end{aligned}
$$

□

Proof of Lemma 4.4.18. For the duration of the proof, set $m' = 2m - 1$. We first increase the dimension of d and d' from $m + 1$ to $m' + 1 = 2m$, by padding them with trailing zeros. Namely, we let $d = (d_0, \ldots, d_m, d_{m+1} = 0, \ldots, d_{m'} = 0)$ and $d' = (d'_0, \ldots, d'_m, d'_{m+1} = 0, \ldots, d'_{m'} = 0)$ (both vectors are now in $[0, 1]^{m'+1}$). For $j \in (m')$, let $a_j = \sum_{i \in (j)}(d_i - d'_i)$ and let $a = (a_0, \ldots, a_{m'})$ (note that $a_j = 0$ for $m \leq j \leq m'$). Also we let $a_j = 0$ for $j \notin (m')$ (in particular, $\mathsf{ES}(a_{\leq j}) = 0$ for $j < 0$). The following claim characterizes the difference $\mathsf{ES}(d'_{\leq j}) - \mathsf{ES}(d_{\leq j})$ in terms of the vector a.

Claim 4.4.20. *For every $j \in (m')$ it holds that $\mathsf{ES}(d'_{\leq j}) - \mathsf{ES}(d_{\leq j}) = (2^\epsilon - 1) \cdot \mathsf{ES}(a_{\leq j-1}) - 2^{j\epsilon} a_j$.*

Proof. An intuitive proof is as follows. Consider the process that changes $d_{\leq j}$ into $d'_{\leq j}$ by "pushing" the amount a_i's from d_i to d_{i+1} for every $i \in (j)$. The effect of these

changes for $i < j$ on $\mathsf{ES}(d'_{\leq j}) - \mathsf{ES}(d_{\leq j})$ is equal to $-a_i 2^{i\epsilon} + a_i 2^{(i+1)\epsilon} = (2^\epsilon - 1)a_i 2^{i\epsilon}$ (a_i is removed from d_i and is added to d_{i+1}, where these changes get multiplied by $2^{i\epsilon}$ and $2^{(i+1)\epsilon}$ respectively in the exponential sums). For $i = j$ the change to $\mathsf{ES}(d'_{\leq j}) - \mathsf{ES}(d_{\leq j})$ is just the negative part $-a_i 2^{i\epsilon}$ (a_i is "pushed out" of $(d_1, \ldots, d_i)$). Formally, the proof goes as follows.

$$
\begin{aligned}
&(2^\epsilon - 1) \cdot \mathsf{ES}(a_{\leq j-1}) - a_j 2^{j\epsilon} \\
&= \big(\sum_{i \in (j-1)} (2^\epsilon - 1) a_i 2^{i\epsilon} \big) - a_j 2^{j\epsilon} \\
&= \big(\sum_{i \in (j-1)} (2^\epsilon - 1) 2^{i\epsilon} \cdot \sum_{k \in (i)} (d_k - d'_k) \big) - 2^{j\epsilon} \cdot \sum_{k \in (j)} (d_k - d'_k) \\
&= \big(\sum_{k \in (j-1)} (d_k - d'_k)((2^\epsilon - 1) \cdot \sum_{k \leq i \leq j-1} 2^{i\epsilon}) \big) - \sum_{k \in (j-1)} (d_k - d'_k) 2^{j\epsilon} - (d_j - d'_j) 2^{j\epsilon} \\
&= \big(\sum_{k \in (j-1)} (d_k - d'_k)((2^\epsilon - 1) \cdot \frac{2^{j\epsilon} - 2^{k\epsilon}}{(2^\epsilon - 1)}) \big) - \sum_{k \in (j-1)} (d_k - d'_k) 2^{j\epsilon} - (d_j - d'_j) 2^{j\epsilon} \\
&= \sum_{k \in (j-1)} (d_k - d'_k)(2^{j\epsilon} - 2^{k\epsilon} - 2^{j\epsilon}) - (d_j - d'_j) 2^{j\epsilon} \\
&= \sum_{k \in (m)} (d_k - d'_k)(-2^{k\epsilon}) = -(\mathsf{ES}(d_{\leq j}) - \mathsf{ES}(d'_{\leq j})) = \mathsf{ES}(d'_{\leq j}) - \mathsf{ES}(d_{\leq j}).
\end{aligned}
$$

□

Note that the second promise of Lemma 4.4.18 implies that $\mathsf{ES}(d'_{\leq j}) - \mathsf{ES}(d_{\leq j})$ can not be "too large" because it is bounded by $\leq (2^\lambda - 1) \cdot \mathsf{ES}(d_{\leq j})$. The next claim, roughly speaking, asserts that if effect of $(a_1, \ldots, a_{j-1})$ in $\mathsf{ES}(d'_{\leq j}) - \mathsf{ES}(d_{\leq j})$ (which is captured by $\mathsf{ES}(a_{\leq j-1})$) is large enough, then a_j that has a negative effect in $\mathsf{ES}(d'_{\leq j}) - \mathsf{ES}(d_{\leq j})$ should also be large to compensate the effect of $(a_1, \ldots, a_{j-1})$ for $\mathsf{ES}(d'_{\leq j}) - \mathsf{ES}(d_{\leq j})$ and keep it 'small". On the other hand, if a_j is large enough then this in turn keeps $\mathsf{ES}(a_{\leq j})$ large. The claim proves the above intuition for the

normalized exponential sums.

Claim 4.4.21. *The following holds for every $j \in (m')$:*

1. $a_j \geq (1 - 2^{-\epsilon}) \cdot \widetilde{\mathsf{ES}}(a_{\leq j-1}) - \lambda \cdot \widetilde{\mathsf{ES}}(d_{\leq j})$, *and*
2. $\widetilde{\mathsf{ES}}(a_{\leq j}) \geq \widetilde{\mathsf{ES}}(a_{\leq j-1}) - \lambda \cdot \widetilde{\mathsf{ES}}(d_{\leq j})$.

Proof. For every $j \in (m')$, the second promise of Lemma 4.4.18 implies that $\mathsf{ES}(d'_{\leq j}) - \mathsf{ES}(d_{\leq j}) \leq (2^{\lambda} - 1) \cdot \mathsf{ES}(d_{\leq j}) < \lambda \cdot \mathsf{ES}(d_{\leq j})$. Therefore, Claim 4.4.20 yields that $(2^{\epsilon} - 1) \cdot \mathsf{ES}(a_{\leq j-1}) - 2^{j\epsilon} a_j < \lambda \cdot \mathsf{ES}(d_{\leq j})$, and thus by normalizing the exponential sums we have

$$a_j \geq 2^{-\epsilon}(2^{\epsilon} - 1) \cdot \widetilde{\mathsf{ES}}(a_{\leq j-1}) - \lambda \cdot \widetilde{\mathsf{ES}}(d_{\leq j}),$$

which proves the first part of the claim.

The second part of the claim also holds since

$$\begin{aligned}
&\widetilde{\mathsf{ES}}(a_{\leq j}) \\
&= 2^{-\epsilon} \cdot \widetilde{\mathsf{ES}}(a_{\leq j-1}) + a_j && \text{by definition of } \mathsf{ES}(\cdot) \text{ and } \widetilde{\mathsf{ES}}(\cdot) \\
&\geq 2^{-\epsilon} \cdot \widetilde{\mathsf{ES}}(a_{\leq j-1}) + (1 - 2^{-\epsilon}) \cdot \widetilde{\mathsf{ES}}(a_{\leq j-1}) - \lambda \cdot \widetilde{\mathsf{ES}}(d_{\leq j}) && \text{by the first part of Claim 4.4.21} \\
&= \widetilde{\mathsf{ES}}(a_{\leq j-1}) - \lambda \cdot \widetilde{\mathsf{ES}}(d_{\leq j}),
\end{aligned}$$

□

It is clear that $\widetilde{\mathsf{ES}}(d_{\leq j}) \leq 1$. Trivially it then holds that $\sum_{j \in (m')} \widetilde{\mathsf{ES}}(d_{\leq j}) \leq m' + 1$. The following claim strengthens this trivial bound.

Claim 4.4.22. *It holds that $\sum_{j \in (m')} \widetilde{\mathsf{ES}}(d_{\leq j}) \leq 2/(2^{\epsilon} - 1)$.*

Proof.

$$\begin{aligned}
&\sum_{j\in(m')} \widetilde{\mathsf{ES}}(d_{\le j}) \\
&= \sum_{j\in(m')} \frac{1}{2^{j\epsilon}} \cdot \sum_{i\in(j)} d_i 2^{i\epsilon} = \sum_{i\in(m')} \sum_{i\le j\le m'} d_i 2^{(i-j)\epsilon} \\
&= \sum_{i\in(m')} d_i \cdot \sum_{k\in(m'-i)} 2^{-k\epsilon} < \sum_{i\in(m')} d_i \cdot \sum_{k\in(\infty)} 2^{-k\epsilon} = 2^\epsilon/(2^\epsilon-1) < 2/(2^\epsilon-1).
\end{aligned}$$

□

Recall that Claim 4.4.21 informally states that if $\widetilde{\mathsf{ES}}(a_{\le j})$ is large for some j, then for $j' > j$, $a_{j'}$ and $\widetilde{\mathsf{ES}}(a_{\le j'})$ are relatively large as well. Looking from the other direction, since eventually we get to the point that $a_{m'} = 0$, none of the $\widetilde{\mathsf{ES}}(a_{\le j})$'s can be too large. This intuition is formalized in the following claim.

Claim 4.4.23. *For every $j \in (m')$, it holds that $\widetilde{\mathsf{ES}}(a_{\le j}) < 4\lambda/(2^\epsilon - 1)$.*

Proof.

$$\begin{aligned}
&\widetilde{\mathsf{ES}}(a_{\le j}) - 4\lambda/(2^\epsilon-1) \\
&< \widetilde{\mathsf{ES}}(a_{\le j}) - \lambda \cdot \big(\sum_{j\le i<m'-1} \widetilde{\mathsf{ES}}(d_{\le i})\big) - 2\lambda/(2^\epsilon-1) && \text{Claim 4.4.22} \\
&\le \widetilde{\mathsf{ES}}(a_{\le m'-1}) - 2\lambda/(2^\epsilon-1) && \text{induction over Claim 4.4.21} \\
&\le \widetilde{\mathsf{ES}}(a_{\le m'-1}) - \frac{\lambda 2^\epsilon}{(2^\epsilon-1)} \cdot \widetilde{\mathsf{ES}}(d_{\le m'}) && \text{by } 2^\epsilon < 2 \text{ and } \widetilde{\mathsf{ES}}(d_{\le m'}) \le 1 \\
&\le \widetilde{\mathsf{ES}}(a_{\le m'-1}) - \frac{\lambda}{(1-2^{-\epsilon})} \cdot \widetilde{\mathsf{ES}}(d_{\le m'}) \\
&\le a_{m'}/(1-2^{-\epsilon}) = 0. && \text{the first part of Claim 4.4.21}
\end{aligned}$$

□

Recall that the conclusion of Lemma 4.4.18 states that if $\overrightarrow{\mathsf{W1}}(d,d') = \sum_{a_i>0} a_i$ is large, then $\overleftarrow{\mathsf{W1}}(d,d') = -\sum_{a_i<0} a_i$ is large too. In Claim 4.4.23, we showed that $\widetilde{\mathsf{ES}}(a_{\leq j})$'s cannot be too large. Roughly speaking (see the calculation below), large $\sum_{a_i>0} a_i$ makes $\widetilde{\mathsf{ES}}(a_{\leq j})$ large, where large $-\sum_{a_i<0} a_i$ makes $\widetilde{\mathsf{ES}}(a_{\leq j})$ small. Thus, in order for the claimed bound on $\widetilde{\mathsf{ES}}(a_{\leq j})$ to hold, $-\sum_{a_j<0} a_j$ should cancel $\sum_{a_j>0} a_j$. Formally, we first show that:

$$\sum_{j\in(m')} \widetilde{\mathsf{ES}}(a_{\leq j}) = \sum_{j\in(m')} \sum_{i\in(m)} a_i 2^{(i-j)\epsilon} = \sum_{i\in(m')} a_i \cdot \sum_{k\in(m'-i)} 2^{-k\epsilon}$$

Therefore from Claim 4.4.23 and the fact that $m' = 2m-1$ we conclude that

$$\sum_{i\in(m')} a_i \cdot \sum_{k\in(m'-i)} 2^{-k\epsilon} \leq 4m'\lambda(2^\epsilon - 1) < 8m\lambda(2^\epsilon - 1) \tag{4.4.14}$$

On the other hand, since we assumed $m\epsilon \geq 1$, for every $0 \leq i < m$ we can get the following upper and lower-bounds for $\sum_{k\in(m'-i)} 2^{-k\epsilon}$ (i.e. the coefficient of a_i in Equation 4.4.14):

$$\frac{1}{2(2^\epsilon-1)} < \frac{(2^\epsilon - 2^{-m\epsilon})}{(2^\epsilon-1)} = \sum_{k\in(m)} 2^{-k\epsilon} \leq \sum_{k\in(m'-i)} 2^{-k\epsilon} < \sum_{k\in(\infty)} 2^{-k\epsilon} = \frac{2^\epsilon}{2^\epsilon-1} < \frac{2}{(2^\epsilon-1)} \tag{4.4.15}$$

By substituting the coefficient of the a_i's in Equation 4.4.14 with the proper upper and lower-bound of Equation 4.4.15 (the positive a_i's with $1/2(2^\epsilon-1)$ and the negative ones with $2/(2^\epsilon-1)$), we get that $\frac{1}{2(2^\epsilon-1)}\sum_{a_j>0} a_j + \frac{2}{(2^\epsilon-1)}\sum_{a_j<0} a_j < 8m\lambda/(2^\epsilon-1)$, which yields that

$$\sum_{a_j>0} a_j + 4 \cdot \sum_{a_j<0} a_j < 16m\lambda. \tag{4.4.16}$$

We conclude that:

$$\overrightarrow{\mathsf{W1}}(d,d') = \frac{1}{m}\cdot\sum_{a_j>0} a_j \leq 16\lambda - \frac{4}{m}\cdot\sum_{a_j<0} a_j = 16\lambda + 4\cdot\overleftarrow{\mathsf{W1}}(d,d').$$

□

Proof of Lemma 4.4.6

Proof. The Protocol VerEmpLabel is defined as follows.

Protocol 4.4.24. $\mathsf{VerEmpLabel} = (\mathsf{P}_{\mathsf{VEL}}, \mathsf{V}_{\mathsf{VEL}})$**.**

Common input: An efficiently decidable set $\mathcal{S} \subseteq \{0,1\}^n$*, an efficiently computable function* $f : \{0,1\}^n \to \{0,1\}^*$*, a claimed size* s *for* $\mathcal{S}$*, the histogram parameter* ϵ*, a claimed histogram* h *of* f *over* $\mathcal{S}$*, the empirical samples* $x_1, \ldots, x_\ell$*, and claimed bins for the empirical samples* u*.*

Description:

1. $\mathsf{V}_{\mathsf{VEL}}$ *verifies that* $x_1, \ldots, x_\ell \in \mathcal{S}$*.*
2. *(Preimage tests) The parties interact for every* $i \in [\ell]$*, in parallel, in an execution of the Set Lower-Bound protocol of Lemma 4.3.12, on input* $(f^{-1}(f(x_i)), s \cdot 2^{-\epsilon\cdot(u(i)+2)}, \epsilon)$*, where* $\mathsf{P}_{\mathsf{VEL}}$ *and* $\mathsf{V}_{\mathsf{VEL}}$ *play the role of* $\mathsf{P}^i_{\mathsf{LB}}$ *and* $\mathsf{V}^i_{\mathsf{LB}}$ *respectively.*
3. *Let* $h^u = \mathsf{Hist}(u)$ *(see Definition 4.3.2). The verifier rejects if* $\mathsf{W1}(h, h^u) > \epsilon$ *(and accepts if not rejected so far).*

Let SetLB^j be the j'th execution of the set Lower-bound protocol in a random execution of VerEmpLabel. In the following we prove the completeness and soundness properties of Protocol 4.4.24.

Completeness. Suppose that the prover is honest (namely $u = u^f$ and $h = h^f$), it follows that

- $\mathsf{V}_{\mathsf{VEL}}$ always accepts in Step 1.
- Proposition 4.3.3 yields that $|f^{-1}(f(x_i))| \geq |\mathcal{S}| \cdot 2^{-\epsilon\cdot(u(i)+1)}$, and the promise on s yields that $|\mathcal{S}| \geq 2^{-\epsilon}s$. Therefore we have $|f^{-1}(f(x_i))| \geq s \cdot 2^{-\epsilon\cdot(u(i)+2)}$ and by the completeness of Lemma 4.3.12 $\mathsf{V}^i_{\mathsf{LB}}$ accepts with probability at least $1-2^{-n}$. Hence $\mathsf{V}_{\mathsf{VEL}}$ does not reject in Step 2 with probability at least $1 - m2^{-n}$.
- $\mathsf{V}_{\mathsf{VEL}}$ always accepts in Step 3.

Therefore $\mathsf{V}_{\mathsf{VEL}}$ accepts with probability at least $1 - m2^{-n} > 1 - 2^{-n/2}$.

Soundness. We claim that $u^f(i) \leq u(i)+3$ for all $i \in [\ell]$ or otherwise $\mathsf{V}_{\mathsf{VEL}}$ rejects with probability at least $1 - 2^{-n}$. Let assume that $u^f(i) \geq u(i) + 4$ for some $i \in [\ell]$. Then Proposition 4.3.3 yields that $|f^{-1}(f(x_i))| \leq |\mathcal{S}| \cdot 2^{-\epsilon\cdot u^f(i)} \leq |\mathcal{S}| \cdot 2^{-\epsilon\cdot(u(i)+4)}$. Now since $|\mathcal{S}| \leq s \cdot 2^{\epsilon}$, it follows that

$$\left|f^{-1}(f(x_i))\right| \leq 2^{-\epsilon}(s \cdot 2^{-\epsilon\cdot(u(i)+2)}) \tag{4.4.17}$$

Hence, by the soundness of Lemma 4.3.12, $\mathsf{V}^i_{\mathsf{LB}}$ (and thus $\mathsf{V}_{\mathsf{SWS}}$) rejects with probability least $1 - 2^{-n}$, in which case we are done.

So in the following we assume that $u^f(i) \leq u(i) + 3$ for all $i \in [\ell]$, which by Proposition 4.3.7 yields that

$$\overleftarrow{\mathsf{SH}}(u^f, u) \leq 3/m. \tag{4.4.18}$$

Using the promise $\mathsf{W1}(h^{u^f}, h^f) \leq \epsilon$ and that $\mathsf{W1}(h^u, h) \leq \epsilon$ (since otherwise $\mathsf{V}_{\mathsf{VEL}}$ would reject in Step 3), we conclude that

$$
\begin{aligned}
&\mathsf{SH}(u^f, u) && (4.4.19)\\
&= \overleftarrow{\mathsf{SH}}(u^f, u) + \overrightarrow{\mathsf{SH}}(u^f, u) && \text{Definition 4.3.6}\\
&\leq \overleftarrow{\mathsf{SH}}(u^f, u) + (\overleftarrow{\mathsf{SH}}(u^f, u) + \overrightarrow{\mathsf{W1}}(h^{u^f}, h^u)) && \text{Lemma 4.3.8}\\
&\leq 6/m + \overrightarrow{\mathsf{W1}}(h^{u^f}, h^f) + \overrightarrow{\mathsf{W1}}(h^f, h) + \overrightarrow{\mathsf{W1}}(h, h^u) && \text{Equation 4.4.18}\\
& && \text{and Proposition 4.3.5}\\
&\leq 6/m + \mathsf{W1}(h^{u^f}, h^f) + \overrightarrow{\mathsf{W1}}(h^f, h) + \mathsf{W1}(h, h^u) && \text{Definition 4.3.4}\\
&\leq 6/m + \epsilon + \overrightarrow{\mathsf{W1}}(h^f, h) + \epsilon && \text{Promise and Step 3}\\
&\leq 6/m + 2\epsilon + (100\epsilon + 4\overleftarrow{\mathsf{W1}}(h^f, h)) && \text{Promise}\\
&\leq 6/m + 102\epsilon + 4(\overleftarrow{\mathsf{W1}}(h^f, h^{u^f}) + \overleftarrow{\mathsf{W1}}(h^{u^f}, h^u) + \overleftarrow{\mathsf{W1}}(h^u, h)) && \text{Proposition 4.3.5}\\
&\leq 6/m + 102\epsilon + 4(\mathsf{W1}(h^f, h^{u^f}) + \overleftarrow{\mathsf{SH}}(u^f, u) + \mathsf{W1}(h^u, h))) && \text{Lemma 4.3.8}\\
& && \text{and Definition 4.3.4}\\
&\leq 6/m + 102\epsilon + 4(\epsilon + 3/m + \epsilon) && \text{Equation 4.4.18}\\
& && \text{and Step 1 and Step 3}\\
&\leq 18/m + 110\epsilon \leq 111\epsilon. && \text{for } m\epsilon = n \geq 18\\
& && (4.4.20)
\end{aligned}
$$

□

4.5 Applications

In this section we use the sample with size protocol described in Section 4.4 for proving our main result. We first formally defined the interactive sampler, Sam, inspired by

the sampler of Haitner et al. [86].

Definition 4.5.1 (Sam oracle)**.** For $d \in \mathbb{N}$, the randomized stateful oracle Sam_d is defined as follows: on input $(C_1, \ldots, C_i, x)$, where $x \in \{0,1\}^m$ and each C_j is a circuit over $\{0,1\}^m$,

1. Sam_d returns $\perp$ if either
   - $i > d$, or
   - it was not previously asked on $(C_1, \ldots, C_{i-1}, x')$ (for some $x' \in \{0,1\}^m$) and answered with x.
2. Otherwise, Sam_d returns a random element in

$$\mathcal{S}(C_1, \ldots, C_{i-1}, x) := \{x' \in \{0,1\}^m \colon \forall j \in (i-1) \colon C_j(x') = C_j(x)\}\,,$$

   where Sam_d uses fresh randomness for each call.

Given an oracle (random) algorithm A and $x \in \{0,1\}^*$, we let $A^{\mathsf{Sam}_d}(x)$ be the output distribution of A^{Sam_d} on input x (this distribution is induced by the random coins of A and Sam_d). We say that $A^{\mathsf{Sam}_d}(x)$ is *k-adaptive*, for $k \in \mathbb{N}$, if $A(x)$ makes at most k *parallel* calls to Sam_d, where a parallel call consist of arbitrary many different inputs to Sam_d (i.e., $(q_1 = C_{1,1}, \ldots, C_{1,j_1}, x_1), \ldots, q_t = (C_{t,1}, \ldots, C_{t,j_t}, x_t))$).

Given the above definition, we can formally state our main result.

Theorem 4.5.2 (Restating Theorem 4.1.1)**.** *For any $d = O(1)$, let A be an efficient oracle algorithm and let $x \in \{0,1\}^*$. If $A^{\mathsf{Sam}_d}(x)$ is k-adaptive, then there exists an $\mathbf{AM}[O(k)]$ protocol AM–$\mathsf{Sam} = (\mathsf{P}, \mathsf{V})$ whose parties get as input $x \in \{0,1\}^*$ and an accuracy parameter $\delta > 1/\operatorname{poly}(|x|)$, the prover P is in $\in \mathbf{BPP}^{\mathbf{NP}}$, and the following hold:*

Completeness: V *accepts in* $\langle \mathsf{P}, \mathsf{V} \rangle (\delta, x)$ *with probability at least* $1 - \delta$.

Soundness: *For every prover* P^* *it holds that*

$$\mathsf{SD}(A^{\mathsf{Sam}_d}(x), \langle \mathsf{P}^*, \mathsf{V} \rangle_{\mathsf{V}} (\delta, x)) \leq \Pr[\langle \mathsf{P}^*, \mathsf{V} \rangle_{\mathsf{V}} (\delta, x) = \perp] + \delta,$$

where $A^{\mathsf{Sam}_d}(x)$ *denotes the output of* A *on input* x*, and* $\langle \mathsf{P}^*, \mathsf{V} \rangle_{\mathsf{V}} (\delta, x)$ *denotes the output of* V *at the end of the interaction with* P^* *on input* (δ, x) *(equals* $\perp$ *if* V *aborts).*

The above theorem yields the following classification.

Corollary 4.5.3. *Let* A *be a* k*-adaptive efficient oracle algorithm such that* A^{Sam_d} *decides a language* $L \subseteq \{0,1\}^n$ *— for every* $x \in \{0,1\}^n$ *it holds that* $\Pr[A^{\mathsf{Sam}_d}(x) = 1_L(x)] \geq \frac{1}{2} + \delta$ *for* $\delta > 1/\operatorname{poly}(n)$*. Then* $L \in \mathbf{AM}[k] \cap \mathbf{coAM}[k]$*, with provers in* $\mathbf{BPP}^{\mathbf{NP}}$.

Proof. In order to keep the text simple, we assume that A makes no parallel queries. Let $\ell < \operatorname{poly}(|x|)$ be an upper-bound on the running A on inputs of length $|x|$. We consider the following protocol for the emulation of $A^{\mathsf{Sam}_d}(x)$:

Protocol 4.5.4.

AM–$\mathsf{Sam} = (\mathsf{P}, \mathsf{V})$**.**

Common input: An accuracy parameter δ *and* $x \in \{0,1\}^*$.

Description: For $i \in [d]$ *let* $\delta_i = (\delta_{i+1})^c / c \cdot \ell^8$*, where* $\delta_d = \delta / 2\ell$ *and* c *is the constant stated in Corollary 4.4.3.*[10]

1. V *chooses, uniformly at random, random coins for* A*, and initialized a table* Prefix *(initially empty).*

[10] Since d is constant, all these values are inverse polynomials of $|x|$ and $1/\delta$.

2. V *emulates* $A^{\mathsf{Sam}_d}(x)$*, while doing the following each time* A *makes a query* $q = (C_1, \ldots, C_i, x)$ *to* Sam_d*:*

   (a) *If* $i > d$*, or* $i > 1$ *and* $\mathsf{st} = (C_1, \ldots, C_{i-1}, x) \notin \mathsf{Prefix}$*, then* V *returns* $\perp$ *to* A *as the answer of* Sam_d*.*

   (b) *Otherwise,* P *and* V *are engaged in a random execution of protocol* SampWithSize *from Corollary 4.4.3 on input* $(\delta_i, \mathcal{S}(\mathsf{st}), \mathsf{Prefix}(\mathsf{st}), C_i)$,[11] *where* V *and* P *act as the verifier and prover respectively.*

   Let (x', s) *be the output of the verifier in the above execution,* V *stores* $\mathsf{Prefix}(C_1, \ldots, C_i, x') = s$ *and returns* x' *to* A *as the answer of* Sam_d*.*

3. V *rejects if it has rejected in one of the above executions.*

It is clear that the complexity of the above protocol matches the statement of the theorem (recall that the prover in SampWithSize is in $\mathbf{BPP}^{\mathbf{NP}}$). We will prove the completeness and soundness of the protocol using induction. In the following we assume that V never sets $\mathsf{Prefix}(C_1, \ldots, C_i, x) = s$ for $s \notin [(1 \pm \delta_i) \cdot |\mathcal{S}(C_1, \ldots, C_{i-1}, x)|]$ (i.e., we assume a variant of V that aborts if the original verifier is about to store an invalid value). Corollary 4.4.3 yields that by doing that we increase the rejecting probability of V by at most $\delta/2$.

Let $\mathrm{View}^j_{\mathsf{Sam}_d}$ denote the view of A after the j'th query to Sam_d. For a prover P^*, we let $\mathrm{View}^j_{\mathsf{P}^*}$ denote the view of A after the j'th query in the emulation done in $\langle \mathsf{P}^*, \mathsf{V} \rangle (\delta, x)$ (where we set it to $\perp$ is V has rejected). Assume that the following for $j \in (\ell)$:

Completeness: V rejects with probability at most $j\delta/2\ell$ when interacting with P up until and including the j'th emulated query.

[11] Where we view a circuit C with m input wires, as a function over $\{0,1\}^m$.

Soundness: For any (unbounded) prover P^* it holds that $\mathsf{SD}(\mathrm{View}^j_{\mathsf{Sam}_d}, \mathrm{View}^j_{\mathsf{P}^*}) \leq \rho_j + j\delta/2\ell$, where ρ_j is the probability that V rejects in the first j queries of $\langle \mathsf{P}^*, \mathsf{V} \rangle (\delta, x)$.

Since the output of the verifier at the end of the emulation is a function of A's view, the above for $j = \ell$ yields the proof of the lemma. For proving the case $j+1$, fix any non-rejecting view v for the first j steps of A. Since ℓ bounds the domain of the set parameter $\mathcal{S}$ in any query made by $A(x)$, Corollary 4.4.3 yields that the following with respect to the $j+1$ query of $A(x)$:

1. V reject with probability at most $\delta/2\ell$ when interacting with P, and
2. $\mathsf{SD}((\mathrm{View}^{j+1}_{\mathsf{Sam}_d} \mid v), (\mathrm{View}^{j+1}_{\mathsf{P}^*} \mid v)) \leq \rho_v + \delta/2\ell$, where ρ_v is the probability that V reject in the $j+1$ query, conditioned on v.

The completeness for the $(j+1)$ step follows immediately from the above and the induction hypothesis. For the soundness, note that $\rho_{j+1} = \rho_j + (1-\rho_j) \cdot \mathrm{E}_{v \xleftarrow{\mathrm{R}} \mathrm{View}^j_{\mathsf{P}^*}} [\rho_v \mid v \neq \perp]$. Similarly, the triangle inequality yields that

$$\begin{aligned}
&\mathsf{SD}(\mathrm{View}^{j+1}_{\mathsf{Sam}_d}, \mathrm{View}^{j+1}_{\mathsf{P}^*}) \leq \\
&\mathsf{SD}(\mathrm{View}^{j}_{\mathsf{Sam}_d}, \mathrm{View}^{j}_{\mathsf{P}^*}) + (1-\rho_j) \cdot \mathop{\mathrm{E}}_{v \xleftarrow{\mathrm{R}} \mathrm{View}^j_{\mathsf{P}^*}} [\mathsf{SD}((\mathrm{View}^{j+1}_{\mathsf{Sam}_d} \mid v), (\mathrm{View}^{j+1}_{\mathsf{P}^*} \mid v)) \mid v \neq \perp] \\
&\leq \rho_{j+1} + (j+1)\delta/2\ell.
\end{aligned}$$

□

4.5.1 Lower-Bounds on Statistically Hiding Commitments

In this section we use Theorem 4.5.2 to derive a lower-bound on the possibility of basing constant-round statistically hiding commitments on the assumption that $\mathbf{P} \neq$

NP. Statistically hiding commitments are defined in Section 4.5.1. In Section 4.5.1 we show that Sam_d can be used to break any d rounds statistically hiding commitment, and define a reduction from statistically hiding commitments to deciding a language in Section 4.5.1. Finally, we state and prove the result of this section in Section 4.5.1.

Statistically Hiding Commitments

Definition 4.5.5 (Statistically hiding commitments)**.** A *(bit) commitment scheme* $(\text{Send}, \text{Rec})$ is an efficient two-party protocol consisting of two stages.[12] Throughout, both parties receive the security parameter 1^n as input.

Commit. The sender Send has a private input $b \in \{0,1\}$, which she wishes to commit to the receiver Rec, and a sequence of coin tosses r. At the end of this stage, both parties receive as common output a *commitment* z.

Decommit. Both parties receive as input a commitment z. Send also receives the private input b and coin tosses r used in the commit stage. This stage is non-interactive: Send sends a single message to Rec, and Rec either outputs a bit (and accepts) or rejects.

Definition 4.5.6. A commitment scheme $(\text{Send}, \text{Rec})$ is *statistically hiding* if

Completeness. If both parties are honest, then for any bit $b \in \{0,1\}$ that Send gets as private input, Rec accepts and outputs b at the end of the decommitment stage.

Statistical Hiding. For every unbounded strategy Rec^*, the distributions $\text{View}_{\text{Rec}^*}(\text{Send}(0), \text{Rec}^*)$ and $\text{View}_{\text{Rec}^*}(\text{Send}(1), \text{Rec}^*)$ are statistically indistinguishable.

[12] Sice we are interested in lower-bounds, we only present the definition for bit commitments.

Computational Binding. For every efficient Send^*, Send^* succeeds in the following game (breaks the commitment) with negligible probability in n:

- Send^* interacts with an honest Rec in the commit stage, which yields a commitment z.
- Send^* outputs two messages τ_0, τ_1 such that for both $b = 0$ and $b = 1$, Rec on input (z, τ_b) accepts and outputs b.

Sam and Statistically Hiding Commitments

Haitner et al. [86] (following Wee [156]) showed that Sam can be used for breaking any statistically hiding commitment. Since there are slight differences between the definition of Sam considered above and the one considered in [86], we restate their result according to our formulation and sketch its proof.

Lemma 4.5.7 (implicit in [86])**.** *For any d-round statistically hiding commitment* $(\text{Send}, \text{Rec})$*, there exists a deterministic oracle adversary A such that A^{Sam_d} break the binding of* $(\text{Send}, \text{Rec})$ *with save but negligible probability.*

Proof Sketch. We assume without loss of generality that Rec speaks first, and let m be the number of random coins used by Send. We also assume that Send gets its random coins, r, as an additional input (i.e., we view Send's input as a pair $x = (b, r)$, where b is the secret bit of Send).

Let $x_0 := \perp$. In the commit stage, A behaves as follows: given a query q_i from Rec, it queries Sam_d on (C_i, x_{i-1}) to get an answer x_i, where C_i is the following circuit: on input $x \in \{0,1\}^{m+1}$, it outputs the message of $\text{Send}(x)$ on the i'th round, given that Rec's first i messages are $q_1, \ldots, q_i$. Finally, A sends $C_i(x_i)$ to Rec (as its i'th message).

In the decommitment stage, A queries Sam_d n times on (C', x_d), where C' is an arbitrary circuit over $\{0,1\}^n$, to get outputs $\{(b_i, r_i)\}_{i=1,\ldots,n}$. If there exists $i \neq j$ such that $b_i \neq b_j$ then A outputs $((b_i, r_i), (b_j, r_j))$, otherwise A aborts.

The definition of Sam_d yields that each (b_i, r_i) is a *random* valid decommitment. Hence, the statistically hiding property of (Send, Rec) yields that, with save but negligible probability, there exist $b_i \neq b_j$ and A successfully produces decommitments to both 0 and 1.[13] Therefore, A^{Sam_d} breaks the binding of (Send, Rec) with save but negligible probability. □

Black-Box Reductions

We now formally define the notion of black-box reductions from deciding a language to (breaking the binding of) commitment schemes.

Definition 4.5.8 (Black-box reduction)**.** A black-box reduction from deciding a language L to breaking the binding of a commitment protocol (Send, Rec) is an oracle algorithm (Send, Rec) with the following guarantee: given as oracle a deterministic and stateless adversary $\mathcal{O}$ that breaks the binding of (Send, Rec), $R^{\mathcal{O}}$ decides L (i.e., , $\Pr[R^{\mathcal{O}}(x) = 1_L(x)] \geq 1 - 2^{-n}$). We say that R is k-adaptive if it makes k adaptive rounds of queries to its oracle; each round may consist of many queries, but all of the queries in one round can be computed without looking at the oracle responses to any of the other queries in the same round.

On Basing Statistically Hiding Commitments on NP-hardness

Given the above definitions, we can formally state result about reducing the security of statistically hiding commitment on the decidability hardness of a given language.

[13] The statistically hiding property yields that given a random transcript of the commitment, essentially half of Send' possible input pairs that are consistent with the transcript are of the form $(0, \cdot)$, and the other half are of the form $(1, \cdot)$.

Corollary 4.5.9. *Suppose that there exists an efficient k-adaptive black-box reduction R from deciding a language L to breaking the binding of a statistically hiding commitment. Then $L \in \mathbf{AM}[k] \cap \mathbf{coAM}[k]$ with provers in $\mathbf{BPP}^{\mathbf{NP}}$.*

Proof. Let R and $(\mathrm{Send}, \mathrm{Rec})$ be the assumed reduction and statistically hiding commitment respectively. Let A^{Sam_d} be the algorithm guaranteed by Lemma 4.5.7 for breaking the binding of $(\mathrm{Send}, \mathrm{Rec})$. We would like to argue that $R^{A^{\mathsf{Sam}_d}}$ decides L, but the problem is that A^{Sam_d} is randomized and stateful. Nevertheless, the following lemma readily follows from Haitner et al. [86].

Lemma 4.5.10 (implicit in [86])**.** *Let* $(\mathrm{Send}, \mathrm{Rec})$*, R and L be as in Definition 4.5.8. The there exists an efficient oracle algorithm $\widetilde{R}$ such that $\widetilde{R}^{\widetilde{\mathcal{O}}}$ decides L for any randomized and stateful oracle $\widetilde{\mathcal{O}}$, which breaks the binding of* $(\mathrm{Send}, \mathrm{Rec})$ *with save but negligible probability.*

Proof Sketch. We present an efficient algorithm $\widetilde{R}$ and a family of deterministic and stateless oracles $\{\mathcal{O}_\lambda\}$ such that the following hold: 1. with save but negligible probability over the choice of λ, it holds that $\mathcal{O}_\lambda$ breaks the binding of $(\mathrm{Send}, \mathrm{Rec})$, and 2. the execution of $\widetilde{R}^{\widetilde{\mathcal{O}}}(x)$ and $R^{\mathcal{O}_\lambda}$ are statistically close, over the randomness of $\widetilde{R}$, R, $\widetilde{\mathcal{O}}$ and a random choice of λ. Showing that will conclude the proof, since the grantees about $\{\mathcal{O}_\lambda\}$ yields that $R^{\mathcal{O}_\lambda}$ decides L correctly for most λ's, and theretofore $\widetilde{R}^{\widetilde{\mathcal{O}}}$ decided L.

Following [86], we first consider a stateless version $\widehat{\mathcal{O}}$ of $\widetilde{\mathcal{O}}$ that lets the caller hold its state — on each query, the caller provides $\mathcal{O}$ with a state parameter (encoded as string), where at the end of the call $\mathcal{O}$ returns the updated state to the caller (in addition to its original output). We would like to claim that whatever can be done with $\widehat{\mathcal{O}}$ could be done with $\widetilde{\mathcal{O}}$. The problem is, however, that a "user" of $\widehat{\mathcal{O}}$ can get additional power by providing fake states. Following [86], this problem is solved by

letting $\widehat{\mathcal{O}}$ sign its states using information theoretic signature (i.e., the output of a random function that $\widehat{\mathcal{O}}$ keeps in its belly), and verify the validity of the signature in each call. Finally, we let $\mathcal{O}_\lambda$ be the oracle $\widehat{\mathcal{O}}$ whose random coins (including the one used for the signatures) fixed to λ.[14]

Since $\widehat{\mathcal{O}}$ breaks the binding of $(\text{Send}, \text{Rec})$ with save but negligible probability, item 1 holds with respect to $\{\mathcal{O}_\lambda\}$. Moreover, the signature mechanism we employ, tell us that, with save but negligible probability over the choice of λ and the random coins of R, invalid calls made by R (i.e., with fake states) are answered with $\perp$.

On input x algorithm $\widetilde{R}$ emulates $R^{\mathcal{O}_\lambda}(x)$ as follows: it forwards the oracle calls of R to $\widetilde{\mathcal{O}}$ (stripped from the state parameter), and returns $\widetilde{\mathcal{O}}$ answers to R, along with the state of $\widetilde{\mathcal{O}}$ and a signature of the state ($\widetilde{R}$ computes both parameters by itself, where for the signature it simply returns a random string). In addition, if the state given in the call is invalid (was not return by a previous call), $\widetilde{R}$ returns $\perp$ as the answer to the call.[15] Finally, it answers identical queries with identical answers (as a stateless oracle should do).

Assuming that R never gets non $\perp$ answers to invalid queries, the distribution of $\widetilde{R}^{\widetilde{\mathcal{O}}}(x)$ and $R^{\mathcal{O}_\lambda}$ are identical. Thus, item 2 follows by the above observation about the guarantee of the signature mechanism. □

Lemma 4.5.10 yields that $\widetilde{R}^{A^{\mathsf{Sam}_d}}$ decides L. Since A is efficient, it follows that there exists an efficient oracle algorithm R' such that R'^{Sam_d} decides L. Hence, Corollary 4.5.3 yields that $L \in \mathbf{AM}[k] \cap \mathbf{coAM}[k]$ with provers in $\mathbf{BPP}^{\mathbf{NP}}$. □

[14] Since the running time of R is bounded, the size of λ is bounded as well.

[15] Note that $\widetilde{R}$ does not need to use the signature mechanism to ensure validity, since $\widetilde{R}$ is stateful and can keep track on the execution.

Chapter 5

More on NP-hard Cryptography, Randomized Reductions, and Checkability of SAT

5.1 Introduction

In this chapter we study various complexity classes under randomized oracle (i.e., Cook) reductions. Such a reduction is an efficient randomized algorithm with access to an oracle, and the algorithm is allowed to ask the oracle multiple questions, even adaptively (using the oracle's answers to generate new questions). The *closure* of a complexity class $\mathcal{C}$ under randomized oracle reductions, denoted $\mathbf{BPP}^{\mathcal{C}}$, is the class of languages (or, more generally, promise problems) that are decidable using efficient randomized algorithms with access to an oracle solving a problem in $\mathcal{C}$.

The closure of many complexity classes under randomized oracle reductions is poorly understood. For example, the following questions remain completely open (i.e., resolving them in either direction would be consistent with standard conjectures

about complexity classes, such as $\mathbf{P} \neq \mathbf{NP}$):

1. Let **PrSZK** denote the class of promise problems having statistical zero-knowledge proofs. How big is the class $\mathbf{BPP}^{\mathbf{PrSZK}}$? For example, does it contain **NP**-complete languages like SAT?

2. Is it possible to use an algorithm computing Nash equilibrium to solve SAT?

3. Is it possible to base one-way functions on **NP**-hardness?

4. Is it possible to prove that $\mathbf{P} \neq \mathbf{NP}$ implies the existence of a hard-on-average problem in **NP**?

In this chapter, we make the following progress on these questions:

1. $\mathbf{BPP}^{\mathbf{PrSZK}} \subseteq \mathbf{AM} \cap \mathbf{coAM}$. Therefore, SAT $\notin \mathbf{BPP}^{\mathbf{PrSZK}}$ unless $\mathbf{PH} = \mathbf{\Sigma}_2$ [30].

2. If SAT $\in \mathbf{BPP}^{\mathbf{TFNP}}$, then SAT has an instance checker. (Recall that computing Nash equilibrium is in **TFNP** [134]).

3. If there is a black-box reduction basing one-way functions on **NP**-hardness, then SAT has an instance checker.

4. If there is a black-box worst-case to average-case reduction for any relation (or language) $R \in \mathbf{NP}$, then R has a non-uniform instance checker.

Our result about **PrSZK** improves on the previously known facts that **PrSZK** is closed under $\mathbf{NC^1}$ truth-table reductions [146] and that $\mathbf{P}^{\mathbf{PrSZK}} \subseteq \mathbf{AM} \cap \mathbf{coAM}$ [152] (where $\mathbf{P}^{\mathbf{PrSZK}}$ denotes the closure of **PrSZK** under *deterministic* oracle reductions, see Theorem 5.1.3).

Regarding our other results, we use the notion of instance checking (also known as program checking) as defined by Blum and Kannan [24]. Intuitively a problem Π

is checkable if there is an efficient randomized algorithm C that uses any program P that purportedly solves Π, such that for every instance x, C^P either decides x correctly or outputs "I don't know" (with very high probability).

One of the open questions posed by Blum and Kannan [24] was whether SAT has an instance checker. This question has remained open for twenty years, and our results imply that, in order to build a reduction that solves SAT using a **TFNP** oracle or using an inverting oracle for a one-way function, one would along the way come up with an instance checker for SAT. Although there is no widely held belief about whether or not SAT is checkable, nevertheless given that this problem has remained unresolved for over twenty years our theorems can be viewed as evidence that solving SAT using a **TFNP** oracle or using an inverting oracle will be "hard to prove" via standard techniques. We credit Holenstein [96] with the observation that connecting the hardness of various problems to the checkability of SAT is a meaningful statement.

5.1.1 Difficulties with Randomized Reductions

We first illustrate the difficulties of understanding randomized oracle reductions. Let $\mathbf{AM} \cap \mathbf{coAM}$ denote the class of *promise* problems Π such that both Π and $\overline{\Pi}$ have **AM** proof systems. Suppose we try to prove that $\mathbf{BPP}^{\mathbf{AM} \cap \mathbf{coAM}} \subseteq \mathbf{AM} \cap \mathbf{coAM}$. (Actually this claim is false unless $\mathbf{NP} \subseteq \mathbf{coAM}$, but we discuss it anyway as it illustrates well the issues involved.)[1] Since $\mathbf{BPP}^{\mathbf{AM} \cap \mathbf{coAM}}$ is closed under complement, it suffices to prove the statement that for every $\Pi \in \mathbf{BPP}^{\mathbf{AM} \cap \mathbf{coAM}}$, Π is also contained in **AM**.

[1] The claim is false because Even et al. [52] observed that there is even a deterministic oracle reduction A that can decide SAT using only an oracle which decides a promise problem in $\mathbf{NP} \cap \mathbf{co\text{-}NP} \subseteq \mathbf{AM} \cap \mathbf{coAM}$. This implies that $\mathbf{NP} \subseteq \mathbf{BPP}^{\mathbf{AM} \cap \mathbf{coAM}}$. This claim is false precisely because of the issues discussed above.

The natural strategy. For every $\Pi \in \mathbf{BPP}^{\mathbf{AM} \cap \mathbf{coAM}}$, Π is decidable by an efficient oracle algorithm A with access to an oracle solving some $\Pi' \in \mathbf{AM} \cap \mathbf{coAM}$. To prove $\Pi \in \mathbf{AM}$ the natural idea is to construct an **AM** protocol for Π where on input z, the prover and verifier do the following:

1. The verifier samples random coins ω for A and sends these to the prover.

2. The prover uses ω to emulate the execution of A. The prover answers any oracle queries "$x \in \Pi'$?" that the reduction asks. Since Π' is a promise problem, the prover is allowed to respond "x does not satisfy the promise". The prover then sends back these oracle queries/answers back to the verifier.

3. The verifier checks that, using ω and the oracle answers claimed by the prover, the execution of A is correct.

4. Since $\Pi' \in \mathbf{AM} \cap \mathbf{coAM}$, this means both $\Pi' = (\Pi'_Y, \Pi'_N)$ and its complement $\overline{\Pi'} = (\Pi'_N, \Pi'_Y)$ have **AM** proofs. Therefore, for every query x satisfying the promise, the verifier asks the prover for proofs that $x \in \Pi'_Y$ or $x \in \Pi'_N$, depending on what the prover claimed previously.

5. If all the checks pass, the verifier outputs whatever A outputs, otherwise the verifier rejects.

One could then hopefully use the correctness of the reduction A to prove completeness and soundness of this protocol. However to prove the soundness of such a protocol one must ensure that a cheating prover cannot trick the verifier into accepting NO instances. This is problematic for the following two reasons:

1. **Prover can falsely claim that a query does not satisfy the promise.** Since the prover can respond "x does not satisfy the promise", and since the

verifier cannot check whether or not x satisfies the promise, the cheating prover may possibly respond "x does not satisfy the promise" even on queries x that do satisfy the promise.

2. **Prover can generate responses depending on A's random coins.** There may be queries $x \notin \Pi'_Y \cup \Pi'_N$ such that the prover is able to falsely claim both $x \in \Pi'_Y$ and $x \in \Pi'_N$ without being caught. Namely, when the verifier runs the **AM** proof to verify that $x \in \Pi'_Y$ or $y \in \Pi'_N$, the prover is able to make the verifier accept in both cases. In this case, the prover may choose to claim that x is a YES or NO instance depending on the random coins of A. But this means that we cannot use the correctness of the reduction A to argue that the verifier obtains the correct answer, because the correctness of A only holds with respect to oracles whose responses are *independent* of A's random coins.

Our results overcome these difficulties in two ways. For **PrSZK** we use additional structure of **PrSZK** to force the prover first never to answer "x does not satisfy the promise", and second to answer in a way that is independent of A's random coins. Our results for checkability aim for the more modest goal of constructing an instance checker for Π, which turns out to be easier than constructing an **AM** protocol for Π.

5.1.2 Complexity of Real-Valued Functions Verifiable in AM

Our first main theorem (which will imply our result about **PrSZK** as a corollary) studies the power of randomized oracle reductions that use oracle access to a class of real-valued functions that we call $\mathbb{R}$-**TUAM** (denoting "real-valued total unique **AM**", see Definition 5.2.5). To understand $\mathbb{R}$-**TUAM**, we begin by discussing the well-known class **TFNP** [117]. The class **TFNP** is the following class of search problems: given input x, find y such that (x, y) satisfy a relation R, with the condition

that R is efficiently decidable and R is *total*, namely for every x, there exists y such that $(x, y) \in R$.

TFAM is a natural relaxation of **TFNP**. We still require that R is total, but now we allow R also to be just verifiable in **AM** (and not necessarily decidable in **P**). For functions where in addition $y \in \mathbb{R}$ and y is unique up to some small error, we say that the function is in $\mathbb{R}$-**TUAM** (see Definition 5.3.1).

Although to the best of our knowledge this class of functions is not well-studied, many natural problems can be decided given a $\mathbb{R}$-**TUAM** oracle. For example, consider the problem of Entropy Difference ED, which is complete for the class **PrSZK** [68]. An instance of this problem is a pair of circuits (X_1, X_2) that we think of as samplers: the distribution sampled by the circuit $X : \{0,1\}^m \mapsto \{0,1\}^n$ is given by the output distribution $X_1(U_m)$ on uniform input bits. We write $H(X)$ to denote the Shannon entropy of the distribution sampled by X. A YES instance of ED satisfies $H(X_1) \geq H(X_2) + 1$, while a NO instance satisfies $H(X_2) \geq H(X_1) + 1$. It is clear that being able to approximate the function $f(X_1, X_2) = H(X_1) - H(X_2)$ is sufficient to decide ED. It turns out that the entropy of distributions sampled by circuits can be verified using an **AM** protocol [2, 56] (see Lemma 5.3.4) and therefore this function f is in the class $\mathbb{R}$-**TUAM**.

Our main theorem about $\mathbb{R}$-**TUAM** is the following:

Theorem 5.1.1. $\mathbf{BPP}^{\mathbb{R}\text{-}\mathbf{TUAM}} \subseteq \mathbf{AM} \cap \mathbf{coAM}$.

Application to PrSZK. Theorem 5.1.1 and the above discussion showing that the **PrSZK**-complete problem ED can be decided using an $\mathbb{R}$-**TUAM** oracle imply the following.

Corollary 5.1.2. $\mathbf{BPP}^{\mathbf{PrSZK}} \subseteq \mathbf{AM} \cap \mathbf{coAM}$.

We note that if randomness is not allowed in the oracle algorithm which uses the **PrSZK** oracle then it is easier to show that we are bound to be in $\mathbf{AM} \cap \mathbf{coAM}$.

Theorem 5.1.3 (Vadhan, Theorem 5.4 in [63])**.** *Let* $\Pi = (\Pi_Y, \Pi_N)$ *be such that there exist sets* S_Y, S_N *satisfying* $S_Y \cup S_N = \{0,1\}^*$, $\Pi_Y \subseteq S_Y$, $\Pi_N \subseteq S_N$, $(S_Y, \Pi_N) \in \mathbf{NP}$, $(S_N, \Pi_Y) \in \mathbf{NP}$. *Then it holds that* $\mathbf{P}^{\Pi} \subseteq \mathbf{NP} \cap \mathbf{co\text{-}NP}$.

This result can be proven the following way (this differs from the proof given in [63]). Consider problems Π that "extend" to a **TFNP** search problem in the following way: there exist disjoint S_Y, S_N containing Π_Y, Π_N respectively such that the relation

$$(x, bw) \in R \;\Leftrightarrow\; (b = 1 \wedge (x, w) \in S_Y) \vee (b = 0 \wedge (x, w) \in S_N)$$

is in **TFNP**. This is equivalent to the hypothesis of Theorem 5.1.3, and solving R immediately implies solving Π. Now combined with the fact that $\mathbf{P}^{\mathbf{TFNP}} \subseteq \mathbf{NP} \cap \mathbf{co\text{-}NP}$ one obtains Theorem 5.1.3.

This definition naturally generalizes to problems Π that "extend" to **TFAM** problems. As we argue in Proposition 5.3.5, the **PrSZK**-complete problem entropy difference can be extended to the **TFAM** problem of computing $f(X_1, X_2) = H(X_1) - H(X_2)$. For the same reason that $\mathbf{P}^{\mathbf{TFNP}} \subseteq \mathbf{NP} \cap \mathbf{co\text{-}NP}$, it also holds that $\mathbf{P}^{\mathbf{TFAM}} \subseteq \mathbf{AM} \cap \mathbf{coAM}$ (Vadhan [152] proved this using a proof along the lines of the proof of Theorem 5.1.3 given in [63]). Therefore, one concludes that

$$\mathbf{P}^{\mathbf{PrSZK}} = \mathbf{P}^{\mathrm{ED}} \subseteq \mathbf{P}^{\mathbf{TFAM}} \subseteq \mathbf{AM} \cap \mathbf{coAM}.$$

Proof technique: Theorem 5.1.1 is proven using the strategy of Section 5.1.1. A $\mathbb{R}$-**TUAM** function is total so the prover can never respond that a query does not satisfy the promise. To prevent the prover from answering in a way that depends on the

Problem	Problem is **NP**-hard	Problem in **PrSZK** via Karp reduction	Can construct one-way function
GapSVP_γ	γ any constant [106]	$\gamma \geq \sqrt{n/\log n}$ [65]	$\gamma \geq \sqrt{n/\log n}$
GapCVP_γ	$\gamma = n^{1/\log\log n}$ [49]	$\gamma \geq \sqrt{n/\log n}$ [65]	$\gamma \geq \sqrt{n/\log n}$
GapSIVP_γ	γ any constant [21]	$\gamma \geq \omega(\sqrt{n\log n})$ [137]	$\gamma \geq \omega(\sqrt{n\log n})$

Table 5.1: Hardness of various lattice problems

random coins chosen by the verifier to emulate the $\mathbf{BPP}^{\mathbb{R}\text{-}\mathbf{TUAM}}$ algorithm, the verifier adds some additional noise to the prover's responses to make them "independent" of the random coins.

Application to basing cryptography on lattice problems: much exciting recent work in cryptography is based on the hardness of various lattice problems. The problems most often studied are GapSVP_γ, GapCVP_γ, and GapSIVP_γ, where $\gamma = \gamma(n)$ is an approximation factor. We refer the reader to [120], for example, for precise definitions of these problems.

Here, we simply note that the hardness of these problems depends critically on the value of γ. This is illustrated in Table 5.1.

Understanding the hardness of these problems is partly motivated by the goal of basing cryptography on **NP**-hardness. Namely, one might hope to show that there is some function γ such that say GapSVP_γ is **NP**-hard, and furthermore that we can construct a one-way function based on the hardness of GapSVP_γ. This would be a great breakthrough in cryptography, since all known cryptosystems are based on much stronger assumptions than **NP**-hardness.

[65] proved negative evidence suggesting that this goal is unattainable. They observed that all known constructions of one-way functions from GapSVP_γ require $\gamma(n) > \sqrt{n/\log n}$. [65] then showed that it is unlikely to prove that GapSVP_γ

is **NP**-hard under Karp reductions because since $\text{GapSVP}_\gamma \in$ **PrSZK**, this would imply **NP** $\subseteq$ **PrSZK** $\subseteq$ **coAM** (which is conjectured to be false since it would imply that **PH** $= \Sigma_2$ [30]). However, [65] does not preclude the possibility of proving that GapSVP_γ is **NP**-hard under randomized reductions. [65, 137] provide similar evidence for GapCVP_γ and GapSIVP_γ.

Our result strengthens the negative evidence to also rule out the use of randomized reductions: notice that saying that GapSVP_γ is **NP**-hard under randomized oracle reductions is equivalent to saying that $\mathbf{NP} \in \mathbf{BPP}^{\text{GapSVP}_\gamma}$. Since $\text{GapSVP}_\gamma \in$ **PrSZK** for $\gamma \geq \sqrt{n/\log n}$, our Corollary 5.1.2 implies that proving GapSVP_γ is **NP**-hard under randomized oracle reductions for any $\gamma \geq \sqrt{n/\log n}$ would imply that **NP** $\subseteq$ **coAM**, and is therefore unlikely. On the other hand, for $\gamma(n) < \sqrt{n/\log n}$, it is unknown whether $\text{GapSVP}_\gamma \in$ **coAM** even if we are considering only Karp reductions.

5.1.3 Randomized Reductions, Checkability, and Testability

Although we are able to overcome the difficulties discussed in Section 5.1.1 to prove Theorem 5.1.1 for the case of $\mathbb{R}$-**TUAM**, we do not know how to bypass both difficulties in general. In the following, we will instead *assume* that the cheating prover is a static, fixed (but possibly faulty) oracle. This by definition prevents the prover from responding in a way that depends on the random coins the verifier sends, and in the settings we study the prover does not benefit by answering "x does not satisfy the promise". The interesting point is that even achieving soundness only against such a severely restricted static prover will imply that the language is "checkable".

Checkability was defined by Blum and Kannan [24] and intuitively guarantees the following: C is an instance checker for a problem Π if, when C is given an instance x and a program P, if P decides Π correctly (for all inputs, not just x), then $C(P, x)$

outputs the correct answer for x with high probability, while if P decides x incorrectly then with high probability $C(P, x)$ either outputs "error" or finds the correct answer despite the incorrectness of $P(x)$. Most instance checkers C require only oracle access to P and not the code of P.

It was already observed by Blum and Kannan [24] that Π is checkable (with C using only oracle access to P) if and only if Π and $\overline{\Pi}$ have interactive proofs where the prover answers only queries of the form "$x \in \Pi$?" and soundness is only required to hold against static cheating provers of the kind described above.

An Application of Beigel's Theorem

Theorem 5.1.4 (Beigel, as cited in [24])**.** *Suppose* Π *is decidable by a randomized reduction* A *with oracle access to* Π'*, and conversely* Π' *is decidable by a randomized reduction* A' *with oracle access to* Π*. Then* Π *is checkable if and only if* Π' *is checkable.*

Proof. Let C_Π be a checker for Π. In order to check Π', given the input x, the checker algorithm $C_{\Pi'}$ first runs the reduction $A'(x)$ and for any query like y that $A'(x)$ wants to ask from its R oracle, $C_{\Pi'}$ does the following. $C_{\Pi'}$ runs the checker algorithm $C_\Pi(y)$ over the "oracle" $A^{\mathcal{O}}$. If it leads to reject, C_Π rejects as well, but if it did not reject and returned the output z, $C_{\Pi'}$ safely uses the answer z for the query y and continues running the reduction $A'(x)$. It is easy to see C_Π's completeness and soundness of follow from those of $C_{\Pi'}$ and the definition of reductions A and A'. □

By definition **TFNP** relations are checkable: given an oracle $\mathcal{O}$, check that $(x, \mathcal{O}(x)) \in R$ indeed holds. Because the relation is total, $\mathcal{O}(x)$ can never claim that no solution exists.

TFNP contains important problems such as Nash equilibrium [134]. Megiddo and Papadimitriou [117] observed that $\mathbf{P}^{\mathbf{TFNP}} \subseteq \mathbf{NP} \cap \mathbf{co\text{-}NP}$ and therefore no

TFNP oracle can be used to decide SAT under deterministic oracle reductions unless **NP** = **co-NP**. Consequently, it is unlikely that problems such as finding Nash equilibrium are **NP**-hard under deterministic oracle reductions. Since **TFNP** is trivially reducible to **NP**, as a corollary of Theorem 5.1.4 we observe the following about the hardness of **TFNP**:

Corollary 5.1.5. *If* SAT $\in \mathbf{BPP}^{\mathbf{TFNP}}$*, then* SAT *is checkable.*

As discussed above, one consequence is that if finding Nash equilibrium can be used to solve SAT via a *randomized* oracle reduction, then SAT is checkable. Corollary 5.1.5 can be interpreted as an incomparable version of the theorem of Megiddo and Papadimitriou [117], where our version handles randomized reductions but arrives at a different (weaker) conclusion.

Extending Beigel's Theorem to Testability

So far we have considered worst-case to worst-case reductions, but equally interesting are worst-case to average-case randomized oracle reductions. Such a reduction is an efficient oracle algorithm A that is guaranteed to decide Π correctly (on all inputs) given oracle access to $\mathcal{O}$ that decides Π' on average over an input distribution D, namely when $\Pr_{x \leftarrow D}[\mathcal{O}(x) \neq \Pi(x)] \leq 1/\operatorname{poly}(n)$.

A notion of checkability for average-case problems also exists which following Blum et al. [26] is called testability. We say that C is a tester for Π' with respect to the input distribution D if the following holds: whenever C is given a program P that correctly decides all x then C must also decide all x. If C is given access to P such that $\Pr_{x \leftarrow D}[P(x) \neq \Pi(x)] \geq \delta$ for some error parameter δ, then C outputs "error" with high probability. This can even be extended to a notion of testability for search problems; see Definition 5.2.7 for a formal definition. We remark that we allow C

to run in time depending on the input length, while the related notion of property testing [26, 72] typically considers testing algorithms that run in time depending only on $1/\delta$. Also, typically C will only use oracle access to P.

Our first observation is that Beigel's theorem can be extended to encompass testability:

Theorem 5.1.6 (Extended Beigel's theorem, informal)**.** *Suppose* Π *is decidable by a worst-case to average-case randomized reduction* A *with oracle access to* Π'*, and conversely* Π' *is decidable by a (worst-case to worst-case) randomized reduction* A' *with oracle access to* Π*. Then if* Π' *is testable, then both* Π *and* Π' *are checkable.*

Application to basing one-way functions on NP-hardness. One of the main goals of theoretical cryptography is to base the existence of cryptographic primitives on reasonable assumptions. $\mathbf{P} \neq \mathbf{NP}$ is a minimal assumption, and it would be ideal if one could construct one-way functions from this assumption. Such a result might be proved by giving an efficient reduction A that uses an oracle $\mathcal{O}$ that inverts a (candidate) one-way function in order to decide SAT. However, it has been shown in a series of works [5, 27, 54, 135] that such a hope is most likely false if A is non-adaptive, since it would imply that $\mathbf{NP} \subseteq \mathbf{coAM}$.

We know less about whether or not one can base one-way functions on **NP**-hardness via an *adaptive* reduction. There are certain cryptographic [4, 92, 121] and complexity-theoretic [10, 55, 94] settings where adaptivity in the reduction buys more power, and so it is important to understand whether adaptive reductions basing one-way functions on **NP**-hardness are possible. Brassard [31] showed that one-way *permutations* cannot be based on **NP**-hardness unless $\mathbf{NP} \subseteq \mathbf{coAM}$. Pass [135] showed that if (general) one-way functions can be based on **NP**-hardness, then certain *witness-hiding* protocols do not exist. However Haitner et al. [89] showed that it is

unlikely that known witness-hiding protocols are of the type studied by [135].

Here we show using different techniques that an adaptive reduction that bases the existence of a one-way function on the hardness of SAT would imply that SAT is checkable. The observation is that a one-way function is testable (for a suitable notion of testability for search problems), and therefore we can apply Theorem 5.1.6.

Corollary 5.1.7 (Joint with Thomas Holenstein)**.** *If there is a randomized oracle reduction that uses an oracle $\mathcal{O}$ which inverts a one-way function in order to decide* SAT*, then* SAT *is checkable.*

We note that the proof that was joint with Holenstein was direct and did not go through the formalism of testability developed here.

Application to basing hardness of learning on **NP***-hardness.* It was shown in [6] that if there exists a reduction uses a PAC learning algorithm to decide SAT (in other words, it bases the hardness of PAC learning on **NP**-hardness), then there also exists a reduction that uses an algorithm that inverts auxiliary-input one-way functions [131] into an algorithm that solves SAT. In the same way that inverting a one-way function is testable, so is inverting auxiliary-input one-way functions. Therefore, using Theorem 5.1.6, we can deduce the following:

Corollary 5.1.8. *If there is a randomized oracle reduction that uses an oracle solving the PAC learning problem in order to decide* SAT*, then* SAT *is checkable.*

Application to worst-case to average-case reductions for SAT Bogdanov and Trevisan [27] prove that non-adaptive worst-case to average-case randomized reductions for **NP** imply that $\mathbf{NP} \subseteq \mathbf{co\text{-}NP}/\operatorname{poly}$. The latter implies that $\mathbf{PH} = \Sigma_3$ which is considered implausible. We will consider the same problem but allow adaptive reductions. Using a technique of [54] we observe that:

Theorem 5.1.9. *Every language $L \in \mathbf{NP}$ is testable by a* non-uniform *tester.*

Theorem 5.4.3 states this formally. Combined with Theorem 5.1.6 this will imply the following.

Corollary 5.1.10 (Informal)**.** *Suppose there is a (possibly adaptive) worst-case to average-case oracle reduction for the relation $R \in \mathbf{NP}$ (where R is* not *necessarily* **NP***-complete). Namely the reduction uses an oracle $\mathcal{O}$ solving R on average in order to decide R on all inputs (with high probability). Then R has a non-uniform instance checker.*

As pointed to us by one of the anonymous reviewers, the original Nisan's proof to show that Permanent has a two-prover proof system can be used to eliminate the needed advice in Corollary 5.1.10 for $R = \text{SAT}$ if the worst-case to average-case reduction for R is of a special form. Namely if the language R is downward self-reducible (which is the case for SAT) and also has a worst-case to average-case reduction where the reduction never asks queries y of length $|y| > n$, then R has a *uniform* instance checker (see Observation 5.4.6.)

5.1.4 Randomized vs. Deterministic Reductions

As already noted throughout the introduction, if we restrict our attention to deterministic oracle reductions then all of the results we prove are already known (and indeed in most cases stronger conclusions hold). We remark that, although it is commonly conjectured that $\mathbf{P} = \mathbf{BPP}$ [25, 99, 127, 158], the techniques used to prove derandomization under commonly held hardness assumptions do not necessarily say that for some oracle $\mathcal{O}$, it holds that $\mathbf{P}^{\mathcal{O}} = \mathbf{BPP}^{\mathcal{O}}$. Indeed, there are examples of $\mathcal{O}$ where $\mathbf{P}^{\mathcal{O}} \neq \mathbf{BPP}^{\mathcal{O}}$. Thus one cannot apply the general derandomization theorems above to the previously known results about, say, $\mathbf{P}^{\mathbf{PrSZK}}$ and $\mathbf{P}^{\mathbf{TFNP}}$ to derive our

results. We will argue directly about $\mathbf{BPP}^{\mathbf{PrSZK}}$, $\mathbf{BPP}^{\mathbf{TFNP}}$, etc. without relying on derandomization assumptions.

5.2 Preliminaries

5.2.1 Promise Problems and Relations

A promise (decision) problem $\Pi = (\Pi_Y, \Pi_N)$ consists of two *disjoint* sets $\Pi_Y \cap \Pi_N = \varnothing$ from $\Pi_Y \cup \Pi_N \subset \{0,1\}^*$.

For every promise problem $\Pi = (\Pi_Y, \Pi_N)$ we write $\Pi(x) = 1$ if $x \in \Pi_Y$ and $\Pi(x) = 0$ if $x \in \Pi_N$. A language L is simply a promise problem $(L, \overline{L})$, where $\overline{L}$ is the complement of L. To extend promise decision problems to search problems, we work with *promise relations*.

Definition 5.2.1 (Promise relations)**.** Let $R = (R_Y, R_N)$ where $R_Y \subseteq \{0,1\}^* \times \{0,1\}^*$ and $R_N \subseteq \{0,1\}^* \times \{0,1\}^*$ such that $R_Y \cap R_N = \varnothing$. We call R a *promise relation* and sometimes write for shorthand $(x,y) \in R$ to mean $(x,y) \in R_Y$. We say R is a *standard* relation if $R_Y = \overline{R_N}$.

For any promise relation $R = (R_Y, R_N)$, define $R(x) = \{y \mid (x,y) \in R_Y\}$. A relation R is *total* if for every x, $R(x) \neq \varnothing$. Note that the decision problem for total relations is trivial, but the search problem may still be hard.

In the following, we will consider $\varnothing$ to also denote a special symbol signifying the empty set, so that search algorithms are allowed to output $\varnothing$ to mean that the algorithm cannot find a solution. We will also abuse notation slightly and allow $A(x) \in R(x)$ to be a true statement if $R(x)$ is empty and $A(x)$ outputs the special symbol "$\varnothing$".

We say an algorithm A solves the language L (resp., the relation R) if for all x, it holds that $A(x) = L(x)$ (resp., $A(x) \in R(x)$). If the algorithm A is randomized, the latter should hold with overwhelming probability (for all x's).

Definition 5.2.2 (**NP** Relation)**.** A standard relation $R \subseteq \{0,1\}^* \times \{0,1\}^*$ is an **NP** relation if the set R is accepted by an efficient algorithm, and for all $(x, y) \in R$, $|y| \leq \text{poly}(|x|)$.

Definition 5.2.3 (**TFNP**)**.** A standard relation $R \subseteq \{0,1\}^* \times \{0,1\}^*$ is in the class **TFNP** if R is an **NP** relation and R is total.

By **PrSZK** we denote the promise version of **SZK**. Misusing the notation, when clear from the context, we use **AM** also to denote its promise version.

AM Promise Relations

Definition 5.2.4 (**AM** relations)**.** For a promise relation $R = (R_Y, R_N)$, define the promise problem $M^R = \{M_Y^R, M_N^R\}$ where $M_Y^R = \{(x, y) \mid (x, y) \in R_Y\}$ and $M_N^R = \{(x, y) \mid (x, y) \in R_N\}$. R is an **AM** relation if $M^R \in$ **AM**, and also $\forall (x, y) \in R_Y$ it holds that $|y| \leq \text{poly}(|x|)$.

Definition 5.2.5 (**TFAM**)**.** A promise relation $R = (R_Y, R_N)$ is in **TFAM** if R is a total relation and also an **AM** relation.

Since our definition is for promise problems, instances $(x, y) \notin R_Y \cup R_N$ can behave arbitrarily. It follows immediately from the definitions that **TFNP** $\subseteq$ **TFAM**.

When are AM (promise) relations interesting? In contrast to **NP**, not every **AM** decision problem has an interesting search version. This is because an **AM** decision problem might not have well-defined witnesses. For example, consider the standard Graph Non-Isomorphism (GNI) protocol, where given (G_1, G_2) the verifier

picks random bit b and sends a random permutation of G_b to the prover, who must then respond with b. There is no fixed "witness" of the non-isomorphism, so the natural **AM** relation of Graph Non-Isomorphism is trivial, $((G_1, G_2), y) \in R_{\mathsf{GNI}}$ iff $(G_1, G_2) \in \mathsf{GNI}$, and the value y is ignored. On the other hand, some interesting problems not known to be in **NP** (such as Entropy Difference ED) do have interesting witnesses that are only known to be verifiable using randomized protocols and not deterministically (see Section 5.3).

5.2.2 Checkability and Testability

Definition 5.2.6 (Checkability)**.** A relation R is *checkable* if there exists an efficient randomized oracle algorithm A such that for all oracles $\mathcal{O}$, the following holds.

- Completeness: Suppose for all x it holds that $\mathcal{O}(x) \in R(x)$. Then for all x, $A^{\mathcal{O}}(x) \in R(x)$ with overwhelming probability.
- Soundness: For any x such that $\mathcal{O}(x) \notin R(x)$, $A^{\mathcal{O}}(x)$ with overwhelming probability either outputs some $y \in R(x)$ or outputs a special error symbol $\perp$, which indicates $\mathcal{O}(x)$ may be wrong.

This definition coincides with the definition of [24] for checkability of promise problems Π if one considers the relation $(x, y) \in R$ iff $\Pi(x) = y$, where $y \in \{0, 1\}$.

It is known [24, 58] that for any $k \geq 2$ and any language L, L is checkable if and only if both L and $\overline{L}$ have an interactive proof system with k provers where the provers are asked only L queries.

An average case notion of checkability was defined by Blum et al. [26], which they called "program testing". (In the following we keep the convention that D_n is a distribution over $\{0, 1\}^n$.)

Definition 5.2.7 (Testability)**.** A relation R is *δ-testable* over the ensemble of distributions $D = \{D_n\}$ if there exists an efficient randomized oracle algorithm A such that for all oracles $\mathcal{O}$ the following holds.

- Completeness: If $\mathcal{O}(x) \in R(x)$ for all x, then $A^{\mathcal{O}}(1^n)$ accepts with overwhelming probability.
- Soundness: If $\Pr_{x \leftarrow D_n}[\mathcal{O}(x) \notin R(x)] > \delta$, then $A^{\mathcal{O}}(1^n)$ outputs $\perp$ with overwhelming probability, indicating that $\mathcal{O}$ may not $(1-\delta)$-solve R.

As with checkability, the definition of program testing for decision problems Π follows immediately by considering the relation $(x, y) \in R$ iff $\Pi(x) = y$.

5.2.3 Worst-Case to Average-Case Reductions

Let D_n be a distribution over $\{0,1\}^n$. We say that the ensemble of distributions $D = \{D_n\}$ is *samplable* if there is an efficient randomized algorithm S which the output of $S(1^n)$ is distributed according to D_n. For $\rho = \rho(n)$, we say that an oracle $\mathcal{O}$ ρ-solves the relation R over D if for every n it holds that $\Pr_{x \leftarrow D_n}[\mathcal{O}(x) \in R(x)] \geq \rho$.

Definition 5.2.8 (Worst-case to average-case reductions.)**.** Let $\rho = \rho(n)$, and let D be an ensemble of distributions. We say that the relation R reduces to ρ-solving the relation R' over D if there is an efficient randomized oracle algorithm A such that $\Pr[A^{\mathcal{O}}(x) \in R(x)] = 1 - \mathrm{neg}(n)$ for every $x \in \{0,1\}^n$ whenever $\mathcal{O}$ ρ-solves the relation R' over D.

Definition 5.2.9 (Worst-case to average-inverting reductions)**.** Let

$$f = \{f_n \colon \{0,1\}^n \to \{0,1\}^n\}$$

be a family of functions. We say that the relation R reduces to ρ-inverting f if R reduces to ρ-solving the relation R^f over the ensemble of distributions D^f where $R^f = \{(y, x) \colon f(x) = y\}$ and $D^f_n = f(U_n)$.

Note that in both Definitions 5.2.8 and 5.2.9, the reduction is allowed to ask oracle queries y of larger length $|y| = \text{poly}(|x|)$ where x is the input to the reduction.

5.3 Real-Valued Total Functions

5.3.1 Definitions and Preliminaries

We begin by defining the class of relations $\mathbb{R}$-**TUAM**, which intuitively captures functions $f : \{0,1\}^* \to \mathbb{R}$ such that given (x, y), it is possible to verify using an **AM** protocol that $|y - f(x)|$ is small.

Definition 5.3.1 ($\mathbb{R}$-**TUAM**)**.** A function $f : \{0,1\}^* \to \mathbb{R}$ is in $\mathbb{R}$-**TUAM** (denoting "real-valued total unique **AM**") if for every $\epsilon \geq 1/\text{poly}(n)$, the following relation $R = (R_Y, R_N)$ is in **AM**:

1. $R_Y = \{(x, f(x)) \mid x \in \{0,1\}^*\}$
2. $R_N = \{(x, y) \mid x \in \{0,1\}^*, y \in \mathbb{R} \; s.t. \; |y - f(x)| > \epsilon\}$

We let $\mathbf{BPP}^{\mathbb{R}\text{-}\mathbf{TUAM}}$ denote the class of promise problems that are decidable by a randomized oracle algorithm given oracle access to a real-valued function whose output is verifiable in **AM**, formalized as follows.

Definition 5.3.2 ($\mathbf{BPP}^{\mathbb{R}\text{-}\mathbf{TUAM}}$)**.** $\Pi \in \mathbf{BPP}^{\mathbb{R}\text{-}\mathbf{TUAM}}$ if there exists an $f \in \mathbb{R}$-**TUAM**, an oracle algorithm A, and an $\epsilon(n) = 1/\text{poly}(n)$ such that for all oracles $\mathcal{O}$ satisfying $\forall x, \; |f(x) - \mathcal{O}(x)| \leq \epsilon(|x|)$, it holds for all $z \in \Pi_Y \cup \Pi_N$ that $A^{\mathcal{O}}(z) = \Pi(z)$ with overwhelming probability over the random coins of A.

Our main interest in studying $\mathbb{R}$-**TUAM** is its relationship to **PrSZK**. We will characterize **PrSZK** by its complete problem: Entropy Difference ED. See e.g., [153] for an introduction and other definitions of **PrSZK**.

Definition 5.3.3 (**PrSZK** and ED, [68])**.** A promise problem Π is in **PrSZK** if and only if it is Karp-reducible to the following promise problem $\mathrm{ED} = (\mathrm{ED}_Y, \mathrm{ED}_N)$. Instances of ED are pairs of circuits (X_1, X_2) where each circuit $X : \{0,1\}^m \mapsto \{0,1\}^n$ is identified by its output distribution $X(U_m)$. Let $H(X) = H(X(U_m))$ denote the Shannon entropy of $X(U_m)$. Then:

1. $(X_1, X_2) \in \mathrm{ED}_Y$ iff $H(X_1) \geq H(X_2) + 1$.

2. $(X_1, X_2) \in \mathrm{ED}_N$ iff $H(X_2) \geq H(X_1) + 1$.

We first recall that the entropy can be estimated using an **AM** protocol.

Lemma 5.3.4 ([2, 56, 68])**.** *For every $\epsilon > 1/\operatorname{poly}(n)$, there is an* **AM** *protocol that on input (X, y) accepts if $H(X) = y$ and rejects if $|H(X) - y| > \epsilon$.*

Proof. We give a simple proof for completeness. Recall the following facts. For two circuits Y_1 and Y_2 sampling distributions, let their concatenation Y_1Y_2 be the joint circuit sampling the product distribution. For all $\epsilon > 0$, it is possible given any $y > 0$ to construct in $\operatorname{poly}(n, 1/\epsilon)$ time a circuit Z_y such that $|H(Z_y) - y| < \epsilon/100$.

It is known that ED can be reformulated so that the gap between X_1 and X_2 is $\epsilon/2$ rather than 1, and still the problem remains complete for **PrSZK**, so let us assume this. Since $\mathrm{ED} \in \mathbf{PrSZK} \subseteq \mathbf{AM}$ [2, 68], therefore given input (X, y), the verifier can run in parallel the **AM** protocol for ED on inputs (XZ_1, Z_y) and (Z_{y+1}, X) and accept iff both executions accept. □

Proposition 5.3.5. $\mathbf{BPP}^{\mathbf{SZK}} \subseteq \mathbf{BPP}^{\mathbb{R}\text{-}\mathbf{TUAM}}$

Proof. We show how to implement an ED oracle using a $\mathbb{R}$-**TUAM** oracle for any $\epsilon < 1$. Consider the function $f(X_1, X_2) = H(X_1) - H(X_2)$. This function is in $\mathbb{R}$-**TUAM** because by Lemma 5.3.4 the entropy of a circuit can be approximated using an **AM** protocol.

We claim that given any oracle $\mathcal{O}$ that solves (the search problem of) the relation R with up to ϵ error, one can decide ED: given (X_1, X_2), query (X_1, X_2) from $\mathcal{O}$ to get $y = \mathcal{O}(X_1, X_2)$ and accept iff $y > 0$. Since we are guaranteed that $|(H(X_1) - H(X_2)) - y| \leq \epsilon$, it therefore holds that if $H(X_1) > H(X_2) + 1$ then $y > 1 - \epsilon > 0$ and if $H(X_1) < H(X_2) - 1$ then $y < -1 + \epsilon < 0$. □

5.3.2 Power of $\mathbb{R}$-TUAM

We now prove Theorem 5.1.1, namely $\mathbf{BPP}^{\mathbb{R}\text{-}\mathbf{TUAM}} \subseteq \mathbf{AM} \cap \mathbf{coAM}$.

Proof of Theorem 5.1.1. Fix any $\Pi \in \mathbf{BPP}^{\mathbb{R}\text{-}\mathbf{TUAM}}$, then by definition there exists $f \in \mathbb{R}$-**TUAM**, an efficient oracle algorithm A, and a parameter $\epsilon \geq 1/\operatorname{poly}(n)$ satisfying Definition 5.3.2. We will follow the natural strategy outlined in Section 5.1.1, but in order to make it work we need to overcome the two difficulties outlined there. The first difficulty is overcome simply because $\mathbb{R}$-**TUAM** is total, and therefore the prover can never respond that a query x does not satisfy the promise. To overcome the second difficulty, we will exploit the fact that $\mathbb{R}$-**TUAM** is a *unique* relation. By adding some noise to the prover's responses, which we can check is close to the unique true answer using the **AM** proof for the $\mathbb{R}$-**TUAM** relation, we will prevent it from making its answers dependent on the verifier's random coins. We now explain how this is done.

Define the (randomized) oracle $\mathcal{O}_\epsilon$ as follows. $\mathcal{O}_\epsilon(x)$ first chooses a uniformly random $\alpha_x \xleftarrow{\mathrm{R}} [-\epsilon/2, +\epsilon/2]$ which we call the "randomizer" of the query x and takes

$y = f(x)+\alpha_x$. Then $\mathcal{O}_\epsilon(x)$ will round y and output $\lfloor y \rceil_\epsilon$ where $\lfloor y \rceil_\epsilon$ denotes the integer multiple of ϵ that is closest to y. Note that it always holds that $\mathcal{O}_\epsilon \in [f(x) \pm \epsilon]$, and so by the definition of A and $\mathcal{O}_\epsilon$, it holds that

$$\Pr_{\mathcal{O}_\epsilon, A}[A^{\mathcal{O}_\epsilon}(z) = \Pi(z)] \geq 1 - n^{-\omega(1)} \tag{5.3.1}$$

for all $z \in \Pi_Y \cup \Pi_N$ (where $\Pi(z) = 1$ if $z \in \Pi_Y$ and $\Pi(z) = 0$ if $z \in \Pi_N$). The reason is that one can choose and fix the randomness of $\mathcal{O}_\epsilon$ first and then Inequality 5.3.1 holds by the definition of A. In the following, let $p(n) = \text{poly}(n)$ be an upper-bound on the number of oracle queries made by A and let ω denote random coins used to run A. Without loss of generality we assume that A does not ask any query x more than once. We also write $\mathcal{O}_\epsilon(x)$ more explicitly as $\mathcal{O}_\epsilon(x, \alpha_x)$ to denote the value of the randomizer α_x used for the answer to the query x.

To prove that $\Pi \in \mathbf{AM}$ it suffices to show an **AM** protocol where either the verifier catches the prover cheating and rejects or (if the prover is honest) the output of the verifier is statistically close to the output of $A^{\mathcal{O}_\epsilon}$. This way the verifier either catches the cheating prover or gets a good emulation of $A^{\mathcal{O}_\epsilon}$ which she can use to take her final decision and choose to accept or reject.

The intuition. The idea is that the verifier will select random coins ω for the execution of A along with real numbers $\alpha_1, \ldots, \alpha_{p(n)}$ drawn uniformly at random from $[\pm\epsilon/2]$ and send these to the prover, and the prover will use ω to run A responding oracle queries according to $\mathcal{O}_\epsilon$ while using α_i as the randomizer for the i'th query x_i. The prover sends back all the $f(x_i)$ to the verifier, who checks for all $i \in [p(n)]$ using $f(x_i)$ and α_i that the responses $\lfloor f(x_i) + \alpha_i \rceil_\epsilon$ give a consistent and accepting execution of the reduction. Furthermore, we assumed that $f \in \mathbb{R}$-**TUAM**, so let R be the corresponding relation guaranteed by Definition 5.3.1. Since R is an **AM**

relation, we will also require that the prover give a proof that all $f(x_i)$'s are correct up to some error δ.

Completeness of this strategy is clear because the honest prover can always prove that $(x, f(x)) \in R$, and so each oracle query x is answered with the same distribution as $A^{\mathcal{O}_\epsilon}$. Soundness follows as well. The reason is that the only time the prover can bias the value $\lfloor f(x_i) + \alpha_i \rceil_\epsilon$ is when $(f(x_i) + \alpha_i \mod \epsilon) \approx \epsilon/2$ for some i, i.e., by slightly perturbing $f(x_i)$ the prover can cause $f(x_i) + \alpha_i$ to be rounded either up or down. But it is easy to see that this bad situation is unlikely, namely for each query, $\Pr_{\alpha_i}[(f(x_i) + \alpha_i \mod \epsilon) \in [\epsilon/2 \pm \delta]] = 2\delta/\epsilon$. By taking $\delta/\epsilon \ll 1/p(n)$, where $p(n)$ is the total number of queries, the verifier gets an emulation of the reduction which is statistically close to an honest emulation and therefore the soundness of the protocol follows from the definition of the reduction.

The AM protocol for the problem Π.

Protocol 5.3.6. *Common input: instance z.*

1. *The verifier V_A sends a random seed ω that will be used to execute A. V_A also sends random numbers $\alpha_1, \ldots, \alpha_{p(n)} \stackrel{R}{\leftarrow} [-\epsilon/2, +\epsilon/2]$.*

2. *The prover P_A emulates the execution of A using random coins ω, where the prover answers the i'th oracle query x_i with $\lfloor y_i + \alpha_i \rceil_\epsilon$. The prover sends back to the verifier the values (x_i, y_i) for all $i \in [p(n)]$. (An honest prover sets $y_i = f(x_i)$.)*

3. *In parallel for all $i \in [p(n)]$, the verifier engages the prover in the* **AM** *protocol that $f(x_i) = y_i$ with approximation error $\delta = \frac{\epsilon}{np(n)}$. If any of these protocols reject, then the verifier rejects.*

4. *The verifier checks for $i = 1, \ldots, p(n)$ that emulating A with the i'th query x_i answered by $\lfloor y_i + \alpha_i \rceil_\epsilon$ leads to A asking the $i+1$'st query x_{i+1}, and so on, and accepts iff A accepts.*

. . . . . .

Since $\mathbf{BPP}^{\mathbb{R}\text{-}\mathbf{TUAM}}$ is closed under complement, proving that the above protocol decides Π suffices to prove Theorem 5.1.1.

Completeness. For any $z \in \Pi_Y$ and for any query x_i, the honest prover computes $y_i = f(x_i)$ and uses $\lfloor y_i + \alpha_i \rceil_\epsilon$ as the response to x_i. Therefore the oracle answers are distributed identically to $\mathcal{O}_\epsilon(x_i, \alpha_i)$, and so by Inequality 5.3.1 the verifier accepts with overwhelming probability.

Soundness. Fix any $z \in \Pi_N$, and let P' be a possibly cheating prover. For each query x_i let y_i be the claim of P' for $f(x_i)$. If for any $i \in [p(n)]$ it holds that $|y_i - f(x_i)| > \frac{\epsilon}{np(n)}$, then with overwhelming probability one of the **AM** protocols in Step 3 will fail by the soundness condition of the $\mathbb{R}$-**TUAM** relation, and so V_A will reject.

So let us suppose that the strategy of P' is restricted to always claim some y_i such that $|y_i - f(x_i)| \leq \frac{\epsilon}{np(n)}$. Now look at the oracle answer $\lfloor y_1 + \alpha_1 \rceil_\epsilon$ used for the first query x_1 in the emulation of the reduction. If it holds that $(f(x_1) + \alpha_1 \mod \epsilon) \notin [\epsilon/2 \pm \delta]$, then for all y_1 satisfying $|y_1 - f(x_1)| \leq \delta$, it holds that $\lfloor y_1 + \alpha_1 \rceil_\epsilon = \lfloor f(x_1) + \alpha_1 \rceil_\epsilon$ and so in this case the prover is unable to control the value of $\lfloor y_1 + \alpha_1 \rceil_\epsilon$. But since α_1 was chosen at random from $[\pm\epsilon/2]$, it holds that:

$$\Pr_{\alpha_1 \xleftarrow{R} [\pm\epsilon/2]} [(f(x_1) + \alpha_1 \mod \epsilon) \in [\epsilon/2 \pm \delta]] = 2\delta/\epsilon \ .$$

It means that with probability $\geq 1-2\delta/\epsilon$ over the choice of $\alpha_1 \xleftarrow{R} [\pm\epsilon/2]$ the prover does not have any control over the first oracle answer used in the emulation of the reduction. The same argument holds for the i'th query for any $i \in [p(n)]$.

It follows by the union bound that the total statistical distance between the set of query/answer pairs generated by $A^{\mathcal{O}_\epsilon}$ and the query/answer pairs generated by (P', V_A) is bounded by at most $\frac{2}{n}$. Therefore, the probability that (P', V_A) accepts is bounded by at most $\leq 2/n + n^{-\omega(1)}$. By repeating the overall protocol in parallel one can reduce this error to be negligible. □

Theorem 5.1.1 and Proposition 5.3.5 yield Corollary 5.1.2, namely $\mathbf{BPP}^{\mathbf{PrSZK}} \subseteq \mathbf{AM} \cap \mathbf{coAM}$.

5.4 Reductions that Imply Checkability of SAT

We saw in Section 5.1.3 that a result of Beigel as cited in [24] (Theorem 5.1.4) along with the fact that **TFNP** is trivially checkable implies the following theorem (see Definition 5.2.3 for a formal definition of **TFNP**).

Theorem 5.4.1. *For any relation $R \in \mathbf{TFNP}$, if* SAT *can be reduced to (the search problem of) R through a randomized reduction, then* SAT *is checkable.*

5.4.1 Extending Beigel's Theorem to Testability

We will prove Theorem 5.4.2 a variant of Theorem 5.1.4 for the case in which one of the reductions in the hypothesis has the extra feature of being worst-case to average case (see Definition 5.2.8). From this stronger assumption we conclude a stronger consequence as follows.

Theorem 5.4.2 (Restating Theorem 5.1.6, formally)**.** *Let $\delta = \delta(n)$, let R and R' be two relations, and let D be an ensemble of distributions. Suppose that solving R reduces to $(1-\delta)$-solving R' over D, and also suppose that there is a randomized (worst-case to worst-case) reduction from R' to R. Then if R' is δ-testable over D (resp., non-uniformly), then R and R' are both checkable (resp., non-uniformly).*

Proof. (of Theorem 5.4.2) We first prove the theorem for the uniform case when there is no advice.

Let A_R be the randomized oracle reduction from solving R to $(1-\delta)$-solving R' over D and let $\ell = \text{poly}(n)$ be an upper-bound on the length of the oracle queries of A_R. Notice that, in particular, A_R is also a worst-case to worst-case reduction from R to R'. Let $A_{R'}$ be the randomized (worst-case to worst-case) reduction from R' to R. By Theorem 5.1.4 and the fact that A_R is also a worst-case to worst-case reduction from R to R', it follows that R is checkable if and only if R' is checkable. Therefore in the following it suffices to show that R is checkable.

One can think of $\mathcal{O}' = A^{\mathcal{O}}_{R'}$ as an oracle which hopefully solves the relation R' correctly. Given a perfect oracle $\mathcal{O}$ for R, by running the reduction $A_{R'}$ with oracle access to $\mathcal{O}$ one can efficiently simulate the oracle $\mathcal{O}'$ which solves the relation R'. Suppose *w.l.o.g.* that $\ell = \text{poly}(n)$ is also an upper-bound on the length of the oracle queries of $A_{R'}$.

Suppose that $T_{R'}$ is a δ-tester for R'. Given the input x, and having access to the oracle $\mathcal{O}$ (which is supposed to solve R') the instance checker $C^{\mathcal{O}}_R(x)$ for the relation R does the following.

- For all $i \in [\ell(n)]$, C_R runs the δ-tester $T_{R'}$ over the oracle $\mathcal{O}' = A^{\mathcal{O}}_{R'}$ to make sure that it decides R' with probability at least $1-\delta$ over D_i for the input length i. $C^{\mathcal{O}}_R(x)$ outputs $\perp$ if $T^{\mathcal{O}'}_{R'}(1^i)$ outputs $\perp$ for any $i \in [\ell]$.

- Otherwise, $C_R(x)$ outputs whatever $A_R^{\mathcal{O}'}(x)$ outputs.

The completeness of C_R is immediate, because if $\mathcal{O}$ is a perfect oracle for R' then for each query y, $A_{R'}^{\mathcal{O}}(y) \in R'(y)$ holds with overwhelming probability. In this case with overwhelming probability $T_{R'}$ will accept also and C_R will output $z \in R(x)$.

The soundness of C_R holds by the soundness of $T_{R'}$ and the definition of A_R. Namely either $\mathcal{O}' = A_{R'}^{\mathcal{O}}$ $(1-\delta)$-solves R' over D_i for $i \leq \ell(n)$, or else $T_{R'}$ outputs $\perp$ with overwhelming probability. In the first case, by the correctness of A_R, the checker C_R gives a correct output with overwhelming probability.

For non-uniform testers, it is clear that the above reduction still holds except one needs to hardwire into C_R the advice strings of $T_{R'}$ for all input lengths $i \leq \ell(n)$. $\square$

Theorem 5.4.3 (Formal statement of Theorem 5.1.9)**.** *Let R be any* **NP** *relation, $\delta = 1/\operatorname{poly}(n)$, and D_n be any ensemble of samplable distributions. Then R can be δ-tested over D_n given $\left\lceil \log \frac{2}{\delta} \right\rceil = O(\log n)$ bits of advice. In particular the advice for length n inputs can be any s_n such that $s_n \leq \Pr_{y \leftarrow D_n}[R(y) \neq \varnothing] \leq s_n + \delta/2$.*

The intuition. Suppose that we are given $s_n \approx \Pr_{y \leftarrow D_n}[R(y) \neq \varnothing]$ (the approximate fraction of YES instances of R over D_n). If we sample enough points $y_1, \ldots, y_k \leftarrow D_n$, then by Chernoff we anticipate roughly a s_n fraction of y_i's to satisfy $R(y_i) \neq \varnothing$. As the first step the checker for R simply verifies that $s_n \approx |\{i \mid \mathcal{O}(y_i) \neq \varnothing\}|/k$ holds. This yet does not mean that with high probability over $y \leftarrow D_n$, $\mathcal{O}(y)$ returns the right answer. But since R is an **NP**-relation we can always make sure that if $\mathcal{O}(y)$ returns $z \neq \varnothing$, then $(y, z) \in R$. By enforcing this extra check over the solutions $z_i = \mathcal{O}(y_i)$, the oracle $\mathcal{O}$ can be wrong only in "one direction": to return $\varnothing$ for a query y that $R(y) \neq \varnothing$. But if $\mathcal{O}$ does so significantly, then it will change its bias $\Pr_{y \leftarrow D_n}[\mathcal{O}(y) \neq \varnothing] \ll s_n$ and it can be detected. A very similar trick is used in [5, 27, 54]. The difference between our setting and [5, 27, 54] is that they deal

with *provers* (rather than oracles) that are stateful and might cheat more intelligently by answering their queries depending on all the queries that they are asked. That makes the job of [5, 27, 54] potentially harder, but they bypass this difficulty by putting strong restrictions on the adaptivity of the reduction (which is not the case for our result). Also [128, 151] use this technique in another setting where the advice $s_n \approx \Pr_{y \leftarrow D_n}[\mathcal{O}(y) \neq \varnothing]$ is used to construct a non-uniform reduction that $(1 - 1/\operatorname{poly}(n))$-solves R over D_n given access to any oracle that solves R correctly only with probability $\geq 1/2 + n^{-1/3+\epsilon}$.

Proof. (of Theorem 5.4.3) Let $\mathcal{O}$ be the oracle that is going to be tested for the relation R. The tester $T_R^{\mathcal{O}}$ acts as follows.

1. Let $k = n/\delta^2$. For $i \in [k]$ sample $x_i \leftarrow D_n$.
2. For $i \in [k]$, ask x_i from $\mathcal{O}$ to get $y_i = \mathcal{O}(x_i)$.
3. If for any $i \in [k]$, it holds that $y_i \neq \varnothing$ but $(x_i, y_i) \notin R$ (which can be checked efficiently), then output $\perp$.
4. If $\frac{|\{i\colon \mathcal{O}(x_i) \in R(x)\}|}{k} < s_n - \frac{\delta}{6}$ then output $\perp$, otherwise accept.

Completeness. If $\mathcal{O}(x) \in R(x)$ for all $x \in \{0,1\}^n$, then T_R never outputs $\perp$ in Step 3. We show that the probability of outputting $\perp$ in Step 4 is negligible. Let $p = \Pr_{x \leftarrow D_n}[R(x) \neq \varnothing]$ and $p' = \frac{|\{i\colon \mathcal{O}(x_i) \in \mathbb{R}(x)\}|}{k}$ be the empirical estimate of p. By Chernoff, it holds that $\Pr_{x_1,\ldots,x_k}[|p' - p| > \epsilon] < 2e^{-2k\epsilon^2}$. Now since it is guaranteed that $s_n \leq p$, therefore it holds that

$$\begin{aligned}\Pr[p' < s_n - \delta/6] &\leq \Pr[p' < p - \delta/6] \leq \Pr[|p' - p| > \delta/6] \\ &< 2e^{-2k(\delta/6)^2} = 2e^{-n/18} = \operatorname{neg}(n).\end{aligned}$$

Soundness. Suppose $\Pr_{x\leftarrow D_n}[\mathcal{O}(x) \notin R(x)] \geq \delta$. Then either it holds that $\Pr_{x\leftarrow D_n}[R(x) = \varnothing \wedge \mathcal{O}(x) \neq \varnothing] \geq \delta/6$ or it holds that $\Pr_{x\leftarrow D_n}[R(x) \neq \varnothing \wedge \mathcal{O}(x) \notin R(x)] \geq 5\delta/6$. We show that in both cases T_R outputs $\perp$ with overwhelming probability.

If $\Pr_{x\leftarrow D_n}[R(x) = \varnothing \wedge \mathcal{O}(x) \neq \varnothing] \geq \delta/6$ then with probability at least $1 - (1 - \delta/6)^k = 1 - \mathrm{neg}(n)$ one of x_i's is sampled such that $R(x_i) = \varnothing$ and $\mathcal{O}(x_i) \neq \varnothing$. In this case $(x_i, \mathcal{O}(x_i)) \notin R$ and so T_R outputs $\perp$ in Step 3.

On the other hand if $\Pr_{x\leftarrow D_n}[R(x) \neq \varnothing \wedge \mathcal{O}(x) \notin R(x)] \geq 5\delta/6$, because $p \leq s_n + \delta/2$ then

$$\begin{aligned}
&\Pr_{x\leftarrow D_n}[\mathcal{O}(x) \in R(x)] \leq \Pr_{x\leftarrow D_n}[R(x) \neq \varnothing \wedge \mathcal{O}(x) \in R(x)] \leq \\
&\Pr_{x\leftarrow D_n}[R(x) \neq \varnothing] - 5\delta/6 = p - 5\delta/6 \leq s_n + \delta/2 - 5\delta/6 \\
&= s_n - \delta/3.
\end{aligned}$$

So for each i it holds that $\Pr_{x_i\leftarrow D_n}[\mathcal{O}(x_i) \in R(x_i)] \leq s_n - \delta/3$, and thus by Chernoff it holds that

$$\begin{aligned}
\Pr[p' \geq s_n - \delta/6] = \Pr[p' \geq (s_n - \delta/3) + \delta/6] &< 2e^{-2k(\delta/6)^2} \\
&= 2e^{-n/18} = \mathrm{neg}(n).
\end{aligned}$$

But if $p' < s_n - \delta/6$, then T_R outputs $\perp$ in Step 4.

□

Worst-Case to Average-Case Reductions for NP

For definitions of worst-case to average-case reductions, see Definition 5.2.8.

Corollary 5.4.4 (Formal statement of Corollary 5.1.10)**.** *Let R be an* **NP***-relation, let $\delta = 1/\mathrm{poly}(n)$ and let D be any efficiently samplable ensemble of distributions. If*

R reduces to $(1-\delta)$*-solving* R *over* D*, then* R *has a non-uniform instance checker.*

Corollary 5.4.5 (Formal statement of Corollary 5.1.7)**.** *Let* $\delta = 1/\operatorname{poly}(n)$*, and let* f *be an efficiently computable family of functions. If* SAT *reduces to* $(1-\delta)$*-inverting* f*, then* SAT *is* uniformly *checkable.*

Proof. (of Corollaries 5.4.4 and 5.4.5) Since SAT is **NP** complete, there is a (worst-case to worst-case) randomized reduction from the **NP** relation R of Theorem 5.4.3 to the relation SAT. Therefore Corollary 5.4.4 follows from Theorem 5.4.2 and Theorem 5.4.3 immediately.

Corollary 5.4.5 also follows from Theorem 5.4.2 and Theorem 5.4.3 similarly by using $R = \text{SAT}$ and using the Cook-Levin reduction (to reduce inverting f to solving a SAT instance). But this time we do not need the advice because of the following. Even though R^f might not be a total relation, but it still holds that

$$\Pr_{y \leftarrow D_i^f}[R^f(y) \neq \varnothing] = \Pr_{y \leftarrow f(U_i)}[y \text{ is invertible}] = 1.$$

Thus here we can use the constants $s_i = 1$ as the advice for all the query lengths $i \leq \operatorname{poly}(n)$ that reduction might ask and apply Theorem 5.4.3, eliminating the need for advice. □

One of the anonymous reviewers indicated to us that for downward self-reducible relations R (which are not necessarily in **NP**) the Corollary 5.4.4 can be improved to get a *uniform* instance checker if the worst-case to average case reduction is of a restricted form:

Observation 5.4.6 (Nisan, observed by an anonymous reviewer)**.** *Let* R *be a downward self-reducible relation, let* $\delta = 1/\operatorname{poly}(n)$ *and let* D *be any efficiently samplable ensemble of distributions. If* R *reduces to* $(1-\delta)$*-solving* R *over* D *by oracle reduction*

A, and if the reduction A never asks queries of length $\ell(n) > n$ from its oracle, then R has a uniform *instance checker.*

Proof. The proof is along the lines of Nisan's proof that Permanent has a two-prover proof system (which is equivalent to a proof system where the prover is stateless and behaves like an oracle).

We show by induction over the input length $k \in [n]$ that R has a δ-tester T_R and an instance checker C_R. By a padding argument we can assume without loss of generality that the worst-case to average-case reduction A always asks queries of the same length as the input length n. Also, if A is the worst-case to average-case reduction, then the following is a checker (as in the proof of Theorem 5.4.2) on inputs of length n: $C_R^{\mathcal{O}}$ first runs T_R to check that $\mathcal{O}$ $(1-\delta)$-solves R on inputs of length n, and if this passes then run $A^{\mathcal{O}}(x)$ and output whatever $A^{\mathcal{O}}$ outputs.

It suffices therefore to construct a tester T_R. Such a tester trivially exists for input length 1. By induction, we define how T_R behaves on input length n assuming we already have a tester for input lengths $n-1$.

1. First run the tester on $\mathcal{O}$ for input length $n-1$, and reject if it rejects.

2. For $i \in [n/\delta^2]$ sample $x_i \leftarrow D_i$ uniformly.

3. For each sampled x_i, run the downward self-reduction of R over x_i which will (perhaps adaptively) generate queries x_{ij} of length $|x_{i,j}| < |x|$ for $j \leq \text{poly}(n)$.

4. Run $A^{\mathcal{O}}$ on each of the queries x_{ij} (which are of length $\leq k-1$).

5. Use the resulting answers along with the downward self-reduction in order to compute the answer for x_i. Call this answer y_i.

6. Reject if there exists any x_i such that either of the following holds:

(a) $\mathcal{O}(x_i) \neq \varnothing$ and $(x_i, \mathcal{O}(x_i)) \notin R$.

(b) $\mathcal{O}(x_i) = \varnothing$ but $(x_i, y_i) \in R$ and $y_i \neq \varnothing$.

Otherwise, accept.

The analysis of the tester T_R is similar to that of Theorem 5.4.2. Let $\mathsf{TIME}_{T_R}(n)$ denote the running time of T_R on inputs of length n. Then it is clear that $\mathsf{TIME}_{T_R}(n) \leq \mathsf{TIME}_{T_R}(n-1) + \text{poly}(n)$, which implies that $\mathsf{TIME}_{T_R}(n) \leq \text{poly}(n)$. The reason is that we only run the tester once for each input length. □

Part III

Unconditional (Statistical) Security

Chapter 6

Zero-Knowledge Interactive PCPs and Unconditional Cryptography

6.1 Introduction

Since the introduction of zero-knowledge proofs in the seminal work of Goldwasser, Micali, and Rackoff [76], a large body of work has been devoted to understanding the capabilities and limitations of such proofs. A particularly successful line of research studied the power of *statistical zero-knowledge* (SZK) proofs — ones which guarantee that even computationally unbounded verifiers can learn nothing from the interaction with the prover. In contrast to computational zero-knowledge proofs [71], a major limitation of SZK proofs which restricts their usefulness in cryptography is that they seem unlikely to cover the entire class of **NP** [2, 56].

Motivated by the above goals, Ben-Or, Goldwasser, Kilian, and Wigderson [15] introduced in 1988 the model of *multi-prover interactive proofs* (MIPs), a natural extension of the standard model of interactive proofs which allows the verifier to interact with two or more non-communicating provers. The main result of [15] is

an unconditional two-prover SZK proof for any language in **NP** (see [9, 50, 113] for subsequent improvements). A direct cryptographic application suggested in [15] is that of proving one's identity using a pair of bank cards. We will further discuss these types of applications later.

In a very surprising turn of events, the initial work on zero-knowledge in the MIP model led to a rapid sequence of developments that have literally transformed the theory of computer science. This line of research culminated in the first proof of the PCP Theorem [7, 8].

The notion of probabilistically checkable proofs (PCPs) is very relevant to our work. In 1988, Fortnow, Rompel, and Sipser [57] suggested an alternative model for MIPs in which multiple provers are replaced by a single oracle, subsequently called a *PCP oracle* or just a PCP. The difference between an oracle and a prover is that an oracle, like a classical proof, cannot keep an internal state. When a prover is asked multiple queries, the answer to each query can depend on all previous queries, whereas the answer of an oracle to each query must depend on that query alone. The latter difference makes soundness against PCP oracles easier to achieve than soundness against provers, which explains the extra power of PCPs over traditional interactive proofs. However, as already observed in [15], the *zero-knowledge* property becomes harder to achieve when converting provers into oracles because oracles have no control over the number of queries made by a dishonest verifier. In particular, if the verifier may query the entire domain of the oracle (as in the case of traditional polynomial-length PCPs) then the oracle can no longer hide any secrets.

The question of replacing zero-knowledge provers by stateless oracles is motivated by practical scenarios in which verifiers can "reset" provers to their initial state, say by cutting off their power supply. (Note that similarly to zero-knowledge provers, zero-knowledge PCP oracles should be *randomized* in the sense that their answer

depends both on the query and on a secret source of randomness which is picked once and for all when the oracle is initialized.) This motivation led to a recent line of work on *resettable zero-knowledge*, initiated by Canetti, Goldreich, Goldwasser, and Micali [39]. The main results from [39] show that, under standard cryptographic assumptions, there exist resettable (computational) zero-knowledge proofs for **NP**. However, results along this line do not seem relevant to the case of *unconditional* (and statistical) zero-knowledge proofs, which are the focus of the present work.

Zero-knowledge PCPs. The question of *unconditional* zero-knowledge PCPs was studied by Kilian, Petrank and Tardos [108] (improving over previous results implicit in [50]). Specifically, it is shown in [108] that any language in **NEXP** admits a proof system with a *single* PCP which is statistical zero-knowledge against verifiers that can make any polynomial number of PCP queries (but are otherwise computationally unbounded). However, as expected from proof systems for **NEXP**, the answers of the PCP oracle cannot be computed in polynomial time. This still leaves hope for scaling down the result to **NP** and making the PCP oracle efficient given an **NP** witness. Unfortunately, such a scaled down version presented in [108] has the undesirable side effect of scaling down the zero-knowledge property as well, effectively restricting the number of queries made by a cheating verifier to be much smaller than the (fixed polynomial) entropy of the oracle. Thus, compared to typical feasibility results in cryptography, the results of [108] for **NP** require us to either make an unreasonable assumption about the computational capability of the (stateless) prover, or to make an unreasonable assumption about the limitations of a cheating verifier.

Interactive PCPs. The above state of affairs motivates us to consider the *Interactive PCP* (IPCP) model, which was recently put forward by Kalai and Raz [102] and further studied in [78]. This model can be seen as lying in between the pure PCP

model and the pure MIP model, thus aiding us in our quest for a "minimal" model for efficient unconditional zero-knowledge proofs for **NP**. In the IPCP model there is one interactive prover as in the MIP model and one PCP as in the PCP model. The study of IPCPs in [102] was motivated by the efficiency goal of allowing *shorter* PCPs for certain **NP** languages than in the traditional PCP model, at the price of a small amount of interaction with a prover. In contrast, our use of the IPCP model is motivated by the *feasibility* goal of obtaining unconditional zero-knowledge proofs for **NP** with polynomial-time prover and PCP oracle. Another difference is that while in the context of [102] a PCP is at least as helpful as a prover, the zero-knowledge property we consider is harder to satisfy with a PCP oracle than with a prover (as discussed above). The IPCP model can be made strictly stronger than the MIP model by requiring soundness to hold also with respect to *stateful* PCP oracles. We tackle this stronger variant as well, but we stick to the basic IPCP model by default.

To meaningfully capture zero-knowledge proofs with polynomial-time provers in the IPCP model, we extend the original IPCP model from [102] in two natural ways. First, we allow the PCP to be randomized. Concretely, we assume that both the prover and the PCP are implemented by polynomial-time algorithms with three common inputs: an instance x, a witness w, and a random input r. (This is analogous to earlier models for efficient multi-prover zero-knowledge proofs for **NP**.) The length of both w and r is polynomial in $|x|$. Second, as discussed above, in order to allow the PCP oracle to hide secrets from the verifier we need to use PCP oracles with a super-polynomial query domain, and we restrict cheating verifiers to make (an arbitrary) polynomial number of queries to the oracle, but otherwise allow them to be computationally unbounded. Note, however, that in contrast to the solutions from [108] we cannot use PCP oracles with a super-polynomial entropy since we want our PCP to be efficiently implementable.

This gives rise to the following feasibility question:

Are there (efficient-prover) statistical zero-knowledge proofs for **NP** *in the interactive PCP model?*

6.1.1 Our Results

We answer the above question affirmatively, presenting an *unconditional* SZK proof for **NP** in the interactive PCP model with efficient prover and PCP oracle. Zero-knowledge holds against cheating verifiers which can make any polynomial (in fact, even sub-exponential) number of PCP queries, but are otherwise computationally unbounded. Our protocol can be implemented in a constant number of rounds. We also show how to get a similar protocol (with a non-constant number of rounds) in the stronger variant of the IPCP model in which a cheating PCP oracle may be stateful, thus strengthening the previous feasibility result from [15].

Interactive locking. The main technical tool we use to obtain the above results (as well as additional applications discussed below) is a new primitive which we call an *interactive locking scheme* (ILS). This primitive extends in a natural way the notion of non-interactive locking schemes which were defined and implemented in [108]. The original locking primitive can be viewed as a PCP-based implementation of a non-interactive commitment with statistical hiding and binding. Roughly speaking, a locking scheme is an oracle which hides a secret that can later be revealed to the receiver by sending it a decommitment key. Given access to the oracle alone, it is hard for the receiver to learn anything about the secret. However, it is easy for the receiver to become convinced that at most one secret can be successfully decommitted even when the oracle is badly formed.

The locking scheme from [108] requires the oracle to have bigger entropy than the number of queries against which the hiding property should hold. We prove the intuitive fact that such a limitation is inherent, and therefore there is no efficient-oracle non-interactive locking scheme which resists an arbitrary polynomial number of queries. This is because intuitively if the entropy of the oracle is bounded, then either: (1) the receiver is able to learn all the entropy by making a polynomial number of queries, and therefore break the hiding property; or (2) if some entropy is hidden no matter what queries the receiver makes, then a cheating sender is able to create a "fake" oracle that can cheat on this entropy and therefore be opened to any value, breaking the binding property.

This motivates our notion of an *interactive* locking scheme. An ILS is a locking scheme in the IPCP model: the commitment phase can involve, in addition to oracle queries by the receiver, interaction with the sender from whom the secret originated. Here the sender and the oracle play the roles of the prover and PCP oracle in the IPCP model, respectively. Decommitment still involves a single message from the sender to the receiver. Somewhat surprisingly (and counter to our own initial intuition), we show that interaction can be used to disrupt the intuitive argument above.

We present several constructions of efficient interactive locking schemes. We show how to obtain such schemes from *interactive hashing* — a primitive which was introduced by Naor, Ostrovsky, Venkatesan, and Yung [125] for the purpose of constructing statistically hiding and *computationally* binding commitment schemes from any one-way permutation (see also [48, 83, 133]). The high level idea of the transformation from interactive hashing to ILS is to "implement" a one-way permutation by an oracle which contains a random *point function* (i.e., a function that outputs 0 on all but one random point). To ensure the binding property even when the oracle is badly formed, the receiver should query the oracle on a small number random points to verify that

it is not "too far" from a point function. The (black-box) proof of security of the interactive hashing protocol implies (unconditional) proof of security for the ILS.

The above connection allows us to use interactive hashing protocols from the literature for obtaining interactive locking schemes, but leaves open the question of minimizing the amount of interaction with the sender. We resolve this question by presenting a novel direct construction of ILS which requires only a single round of interaction with the sender.

The high level idea behind our single round ILS is as follows. The oracle π constructed by the sender will be the zero function over $\{0,1\}^n$ except for an "interval" of size 2^{cn}. That is, $\pi(x) = 1$ for $a \leq x \leq a + 2^{cn}$ and $\pi(x) = 0$ elsewhere. Depending on whether the sender commits to zero or one, the interval will be planted in the first or second half of the oracle π. The position a of the interval will be revealed to the receiver in the decommitment phase. When $c < 1$, the interval size 2^{cn} will be small enough to prevent the receiver from finding the committed bit during the commitment phase. But now the sender is able to cheat by planting intervals in both the first and second half of π. To guarantee binding, we let the receiver ask a "challenge" question about the interval in such a way that the sender cannot find a *pair* of planted intervals in the first and second half of π with the same challenge answer. A natural idea is to use a pairwise independent function $h\colon \{0,1\}^n \to \{0,1\}^{dn}$ and ask the sender to reveal $h(a)$. The sender is able to plant at most $2^{(1-c)n}$ *separate* intervals in each half of π. Each of the intervals in the first and second half of π will have the same hashes with probability 2^{-dn}. Therefore if $2(1-c) < d$, then with high probability over the choice of h the sender is *not* able to find two intervals with the same hash value $h(a)$ and thus gets committed to a fixed bit. But now the information revealed by $h(a)$ might help the receiver find a non-zero point in π and break the hiding property. We show how to modify the a known construction of pairwise independent hash functions to

get another function which is still almost pairwise independent but has the additional property that the preimages of any hash value are "scattered" in the domain of the hash function. The latter property prevents the receiver from taking advantage of the knowledge of $h(a)$ to find where the interval is planted. Using this approach we simultaneously guarantee binding and hiding.

Cryptography using hardware tokens. The above study of zero-knowledge interactive PCPs and interactive locking schemes is motivated by a recent line of research on the capabilities of cryptographic protocols in which parties can generate tamper-proof hardware tokens and send them to each other. Katz [105] shows that, under computational assumptions, general *universally composable* (UC) secure two-party computation [37] is possible in this model if the tokens are allowed to be *stateful*, and in particular can erase their secrets after being invoked. It was subsequently shown that even *unconditional* security is possible in this model, first for the case of commitment [123] and then for general tasks [80]. See [67, 77, 93] and references therein for other applications of stateful tokens in cryptography.

Obtaining similar results using *stateless* tokens turns out to be more challenging. Part of the difficulty stems from the fact that there is no guarantee on the functionality of tokens generated by malicious parties — they may compute arbitrary functions of their inputs and may even carry state information from one invocation to another. It was recently shown in [80], improving on [40], that any *one-way function* can be used for basing (computationally) UC-secure two-party computation on stateless tokens. More practical protocols which satisfy weaker notions of security were given in [110]. These works leave open the question of obtaining a similar result *unconditionally*, and with *statistical* security. (To get around impossibility results in the plain model, the number of queries to a token should be polynomially bounded, but otherwise

malicious parties may be computationally unbounded.) In fact, the constructions from [40, 80, 110] may lead to a natural conjecture that achieving statistical security in this setting is impossible, since in these constructions all the "useful information" contained in tokens can be learned by a computationally unbounded adversary using a polynomial number of queries.

However, similar to the case of ILS discussed above, the combination of stateless tokens and interaction turns out to be surprisingly powerful. As already alluded to in [15], MIP protocols can naturally give rise to protocols in the hardware token model. In our case, we implement the ILS (or IPCP) by having a single sender (prover) create a stateless tamper-proof hardware token which implements the PCP oracle and send it to the receiver (verifier). Applying this to our results, this directly gives rise to the first unconditionally secure commitment protocols and SZK proofs for **NP** using stateless tokens.

We show how this can be extended to general unconditionally secure two-party computation if parties are allowed to build tokens which encapsulate other tokens: namely, the receiver of a token A is allowed to build another token B which internally invokes A. The high level idea is the following. By the completeness of oblivious transfer (OT) [101, 107], it suffices to realize OT using stateless tokens. This is done as follows. The OT sender's input is a pair of strings (s_0, s_1) and the OT receiver's input is a selection bit b. The OT receiver commits b using an ILS. Applying our best construction, this involves sending a token A to the OT sender and responding to a random challenge message received from the OT sender. The OT sender now prepares and sends to the receiver a token B with the following functionality. Token B accepts a selection bit b along with a corresponding decommitment message. It checks that the decommitment is valid (this involves invocations of the token A, which token B encapsulates) and then returns the string s_b if decommitment was successful. The

binding property of the ILS guarantees that the OT receiver can learn at most one string s_b. The hiding property of the ILS guarantees that the sender cannot learn b.

Interestingly, we also show a matching negative result: if token encapsulation is not allowed, then statistically secure OT is impossible. This holds even if both parties are guaranteed to follow the protocol except for making additional queries to tokens in order to learn information about the other party's input. The proof of this negative result employs the recent notion of accessible entropy from [84] and has the following high level intuition. One way to explain why statistical OT is not possible in the standard model (i.e., without tokens) is that at any time during the interaction a computationally unbounded party (e.g., the OT receiver R) is able to sample from the space of its randomness r_R conditioned on what the other party (i.e., the OT sender S) knows about R's computation. The latter information is captured by the transcript τ of the protocol. Therefore if at the end of the interaction the distribution of r_R conditioned on τ is not revealing the receiver's bit b, the receiver can sample from $(r_R \mid \tau)$ and find out both of the strings (s_0, s_1). If such sampling is done efficiently, the protocol is said to have accessible entropy [84]. In the token model, however, the exchanged information is not symmetric and Bob might not know which queries Alice has asked from Bob's tokens. Similarly to the standard model, we define a protocol (A, B) in the token model to have accessible entropy if the parties can (information theoretically) sample their history of computation so far only conditioned on the *other* party's view. Similarly to the standard model, accessing the entropy of a protocol for OT in the token model can be used to break its security for the benefit of either of the parties. We prove the following technical lemma: For any protocol (A, B) in the stateless token model, there is another protocol (A', B') with the following properties. **(1)** the parties in (A', B') "emulate" the original protocol (A, B). Namely A' honestly runs an instance of A in its belly and similarly B' does so for an instance of B. **(2)**

Other than emulating (A, B), A' and B' are also allowed to learn more information about the tokens they hold by asking token queries that might not be specified by (A, B). **(3)** (A', B') is still restricted to asking at most poly(n) number of queries from the tokens totally. **(3)** Finally, almost all the entropy of the interactive algorithms A' and B' in the protocol (A', B') is accessible. This lemma, together with the necessity of inaccessible entropy for the possibility of OT, proves that statistical OT does not exist in the stateless token world.

6.2 Preliminaries

6.2.1 Properties of Interactive and Oracle Algorithms

Although one of the main applications of the interactive protocols is to use them as a "proof system" [76], here we define their basic properties in a more abstract level so that we can use them in the next section in the extended model of Interactive PCPs.

In this chapter we use the notation $\widehat{A}$ to denote a possibly malicious interactive algorithm participating in a protocol instead of the honest interactive algorithm A.

Definition 6.2.1 (Properties of interactive protocols)**.** Let (P, V) be an interactive protocol composed of a prover P and a verifier V. We define the following properties for (P, V).

- **Soundness:** (P, V) is $(1 - \delta)$-sound for the input x, if for every prover $\widehat{P}$ it holds that $\Pr[\left\langle \widehat{P}, V \right\rangle (x) = 1] \leq \delta$.
- **Statistical zero-knowledge (SZK):** (P, V) is $\epsilon(n)$-SZK for the language L if there is a simulator Sim that gets *oracle access*[1] to an arbitrary (unbounded) verifier $\widehat{V}$, runs in time poly(n), and *if* $x \in L$ then $\mathsf{Sim}^{\widehat{V}}$ produces a view for $\widehat{V}$

[1] All the zero-knowledge constructions in this thesis use *black-box* simulators.

which is $\epsilon(n)$-close to the view of $\widehat{V}$ when interacting with P on the input x. A query $q_{\mathsf{sim}} = (r_V, a_1, \ldots, a_i)$ of the simulator Sim to the oracle $\widehat{V}$ consists of the randomness r_V for $\widehat{V}$ and a set of first i messages of the prover $a_1, \ldots, a_k$ to the verifier. The answer $a_{\mathsf{ver}} = (q_1, \ldots, q_{i+1})$ returned by the oracle $\widehat{V}$ consists of the first $i+1$ messages of the verifier $\widehat{V}$ when r_V is used as the randomness and $a_1, \ldots, a_k$ are received from the prover. Thus the simulator Sim, in general, can "rewind" $\widehat{V}$ by asking queries of the form $q_{\mathsf{sim}} = (r_V, a_1, \ldots, a_i)$ and $q'_{\mathsf{sim}} = (r_V, a_1, \ldots, a_{i-1}, a'_i)$. The simulator Sim is called *straight-line* if its queries are of the following "incremental" form: $(r_V), (r_V, a_1), \ldots, (r_V, a_1, \ldots, a_i)$ and the output of $\mathsf{Sim}^{\widehat{V}}$ is the last query of Sim: $(r_V, a_1, \ldots, a_j)$. It is easier to think of a straight-line simulator as an interactive algorithm which simply interacts with the verifier $\widehat{V}$ and outputs the view of $\widehat{V}$ in this interaction.

Note that the soundness is only a property of the verifier V while zero-knowledge is only a property of the prover P.

Inefficient verifiers with large randomness. To prove the zero-knowledge for inefficient verifiers $\widehat{V}$ with super-polynomially long randomness, we slightly change the simulator's behavior of Definition 6.2.1 as follows. We let the oracle $\widehat{V}$ to choose the randomness r_V uniformly at random (hidden from the simulator) and the simulator's queries would only consist of prover's messages $q_{\mathsf{sim}} = (a_1, \ldots, a_i)$. Also the simulator's output will be the last query of the simulator $q_{\mathsf{sim}} = (a_1, \ldots, a_j)$ joint with the randomness r_V chosen by the verifier (and yet $(r_V, a_1, \ldots, a_j)$ should be ϵ-close to the view of $\widehat{V}$ when interacting with P). All of our simulators in this chapter will be of this form to let us handling arbitrarily inefficient verifiers.

Definition 6.2.2 (Composition of interactive protocols)**.** Let $(P_1, V_1), \ldots, (P_k, V_k)$ be k interactive protocols. By $(P_{\mathsf{seq}}, V_{\mathsf{seq}})$ we mean the *sequential composition* of

$(P_1, V_1), \ldots, (P_k, V_k)$ defined as follows.

- Let x be the common input.
- V_{seq} uses independent random coins $r_{V_1}, \ldots, r_{V_k}$ for $V_1, \ldots, V_k$ and P_{seq} uses independent coins $r_{P_1}, \ldots, r_{P_k}$ for $P_1, \ldots, P_k$.
- P_{seq} and V_{seq} first interact according to the protocol (P_1, V_1) over the input x, and after that interact according to the protocol (P_2, V_2) (over the asme input x) and so on.
- At the end V_{seq} rejects if any of V_i's reject.

By $(P_{\mathsf{par}}, V_{\mathsf{par}})$ we mean the *parallel composition* of $(P_1, V_1), \ldots, (P_k, V_k)$, which is defined similar to $(P_{\mathsf{seq}}, V_{\mathsf{seq}})$ with the difference that $V_1, \ldots, V_k$ interact with the prover P_{par} in a *round-synchronized* way. Namely in the j'th round of the interaction, V_{par} sends the queries $q_1^j, \ldots, q_k^j$ to the prover P_{par} where q_i^j is the j'th query of V_i (and $q_i^j = \perp$ if V_j's interaction finishes before the j'th round). Then V_{seq} receives k answers $a_1^j, \ldots, a_k^j$ and V_i uses a_i as its answer to continue its execution.[2]

Both $(P_{\mathsf{seq}}, V_{\mathsf{seq}})$ and $(P_{\mathsf{par}}, V_{\mathsf{par}})$ are special cases of *concurrent composition* $(P_{\mathsf{con}}, V_{\mathsf{con}})$ in which the verifier runs $V_1, \ldots, V_k$ and the prover runs $P_1, \ldots, P_k$ in a way that V_i will be interacting with P_i in the i'th "sessions". There is not any round synchronization enforced across different sessions. The honest verifier V_{con} will send the next query q_i^j (i.e. the j'th query of V_i) when she receives the answer a_i^{j-1} from P_i in the i'th session, but a cheating verifier $\widehat{V_{\mathsf{con}}}$ might *delay* sending such query to gather more answers in different sessions before continuing. On the other hand the honest prover P_{con} will send the answer a_i^j back whenever he receives the query q_i^j from V_{con},

[2]To define the parallel composition, we do not necessarily assume $(P_1, V_1), \ldots, (P_k, V_k)$ to have equal round complexity, and the round complexity of $(P_{\mathsf{par}}, V_{\mathsf{par}})$ will be the maximum of the round complexity of $(P_1, V_1), \ldots, (P_k, V_k)$.

but a cheating prover $\widehat{P_{\mathsf{con}}}$ might delay sending such answer to gather more queries in different sessions before continuing.

By using simple "delay policies", both the prover and the verifier (when the other party is honest) are able to force a concurrent composition to behave like a parallel or sequential composition.

The inputs in composed protocols. In all variants of compositions defined above (including the concurrent composition), the same input x is used in all the sub-protocols. More general definitions of concurrent composition allow different inputs to be used in different instances of the protocol, but our definition still generalizes the parallel and sequential definitions and is enough for our purposes. Moreover, the properties of the concurrent composition described in Lemma 6.2.10 below (in particular the SZK property) hold in the more general form of concurrent composition with different inputs (as long the inputs are fixed at the beginning of the execution of the system.)

Lemma 6.2.3 (Properties of composition of interactive protocols)**.** *Let* $(P_1, V_1), \ldots (P_k, V_k)$ *be* k *interactive protocols, and let* $(P_{\mathsf{seq}}, V_{\mathsf{seq}}), (P_{\mathsf{par}}, V_{\mathsf{par}}), (P_{\mathsf{con}}, V_{\mathsf{con}})$ *be, in order, their sequential, parallel, and concurrent compositions. It holds that:*

1. **Soundness [11]:** *If* (P_i, V_i) *is* $(1-\delta_i)$*-sound over the input* x *for all* $i \in [k]$*, then* $(P_{\mathsf{con}}, V_{\mathsf{con}})$ *(and in particular* $(P_{\mathsf{seq}}, V_{\mathsf{seq}})$ *and* $(P_{\mathsf{par}}, V_{\mathsf{par}})$*) will have soundness* $1 - \prod_i \delta_i$*.*

2. **SZK [111]:** *If* (P_i, V_i) *is* ϵ_i*-SZK for the language* L *with a straight-line simulator for all* $i \in [k]$*, then* $(P_{\mathsf{con}}, V_{\mathsf{con}})$ *(and in particular* $(P_{\mathsf{seq}}, V_{\mathsf{seq}})$ *and* $(P_{\mathsf{par}}, V_{\mathsf{seq}})$*) will be* $(\sum_i \epsilon_i)$*-SZK for* L *with a straight-line simulator.*

Part 1 of Lemma 6.2.3 is proved in [11] for the case of parallel composition, but the proof extends to the concurrent composition as well. The proof of Part 2 of Lemma 6.2.10 follows by a standard hybrid argument similar to the proof of Part 2 in Lemma 6.2.14 below.

For the oracles $\pi_1, \ldots, \pi_k$, by $\pi = (\pi_1 | \ldots | \pi_k)$ we mean their *combined* oracle defined as follows: Given the query (i, q), π answers as $\pi(i, q) = \pi_i(q)$. We allow an oracle algorithm A^π to ask multiple oracle queries from π in one *round* of queries.

By a malicious *stateful* oracle $\widehat{\pi}$ we mean an interactive algorithm whose honest behavior is like an oracle. In other words, given a round of queries $q_1, \ldots, q_k$, π returns $\pi(q_1), \ldots, \pi(q_k)$, but a malicious stateful oracle $\widehat{\pi}$ accessed by the oracle-algorithm A can respond the queries similar to a malicious interactive algorithm interacting with A. That is, $\widehat{\pi}$'s answer to a query q can depend on the common input x, or the previous queries of A asked from $\widehat{\pi}$, or other queries being asked in the same round as q.

Definition 6.2.4 (Parallel composition of oracle algorithms)**.** Let $A_1^{\pi_1}, \ldots, A_k^{\pi_k}$ be k oracle algorithms with access to k oracles $\pi_1, \ldots, \pi_k$ where for all $i \in [k]$ A_i asks at most t rounds of oracle queries from π_i. We define $A_{\mathsf{par}}^{\pi_{\mathsf{par}}}$ the *parallel composition* of $A_1^{\pi_1}, \ldots, A_k^{\pi_k}$ as follows.

- The oracle $\pi_{\mathsf{par}} = (\pi_1 | \ldots | \pi_k)$ is the combined oracle of $\pi_1, \ldots, \pi_k$.
- $A_{\mathsf{par}}^{\pi_{\mathsf{par}}}$ emulates $A_1^{\pi_1}, \ldots, A_k^{\pi_k}$ in parallel as follows. Let q be one of the queries of A_i in its j'th round of queries to π_i. A_{par} asks (i, q) from π_{par} in its j'th round of queries and returns the answer to A_i.
- $A_{\mathsf{par}}^{\pi_{\mathsf{par}}}$ rejects if any of the emulated algorithms $A_1^{\pi_1}, \ldots, A_k^{\pi_k}$ reject.

6.2.2 Interactive PCPs

Interactive PCPs (Definition 6.2.5 below) were first introduced in [102] and combine the notion of oracle algorithms with interactive algorithms. Similar to the previous section, here also we define IPCPs in a general way, not only for the purpose of a proof system, but rather as a model of interaction consisting of interactive algorithms and a prover.

Definition 6.2.5. (Adapted from [102]) An *interactive probabilistically checkable proof* (IPCP) $\Gamma = (P, \pi, V)$ consists of an interactive algorithm P (the *prover*), an oracle π (the *PCP oracle*), and an interactive algorithm V (the *verifier*) such that:

- P and π share common randomness r_P, and V is given the randomness r_V.
- P, π, and V will be given an input x of length $|x| = n$. P and π may also receive a common private input w.[3]
- The PCP oracle π is a function of (r_P, x, w, q) where q is a query of the verifier V. Since (r_P, x, w) is fixed at the beginning of the protocol, we might simply use $\pi(q)$ to denote the answer to the query q.
- P and V^π engage in an interactive protocol during which V can query the PCP oracle π and at the end V accepts or rejects.

By an *efficient* IPCP we mean one in which the prover P, the PCP oracle π, and the verifier V run in polynomial time over the input length $|x| = n$.

Definition 6.2.6 (Properties of IPCPs)**.** Let $\Gamma = (P, \pi, V)$ be an IPCP. We define the following properties for Γ.

[3]For example when (P, π) are efficient and $L \in \mathbf{NP}$, w could be a witness for $x \in L$.

- **Soundness:** Γ is $(1-\delta)$-sound for input x, if for every prover and oracle $(\widehat{P}, \widehat{\pi})$ it holds that $\Pr[\left\langle \widehat{P}, V^{\widehat{\pi}} \right\rangle (x) = 1] \leq \delta$.

- **Adaptive-soundness:** Γ is $(1-\delta)$-adaptively-sound for input x, if for every prover $\widehat{P}$ and every *stateful* oracle $\widehat{\pi}$ it holds that $\Pr[\left\langle \widehat{P}, V^{\widehat{\pi}} \right\rangle (x) = 1] \leq \delta$.

- **Statistical zero-knowledge (SZK):** The SZK property is defined as an extension to the SZK property of interactive protocols (Definition 6.2.1) with respect to a language L. Since all of our simulators in this chapter are straight-line, for sake of simplicity here we only describe how to extend the definition to SZK of IPCPs for straight-line simulators.

  - The straight-line simulator interacts with a (potentially malicious) verifier $\widehat{V}$, while the simulator Sim receives *all* the queries of the the verifier (including both the queries asked from the prover and from the oracle) and responds to them.

  - Since an unbounded verifier can ask arbitrary number of queries from its oracle, here we put a bound u on the number of oracle queries asked by $\widehat{V}$. More formally we say that Γ is $(u(n), \epsilon(n))$-SZK (with a straight-line simulator), if there is a simulator described as above such that for any $v \leq u$, if $\widehat{V}$ asks at most v oracle queries, then Sim runs in time $\mathrm{poly}(n, v)$ and produces a view for $\widehat{V}$ which is ϵ-close to the view of $\widehat{V}$ when interacting with (P, π).

Note that when $u(n)$ is super-polynomial, Definition 6.2.7 implies the standard notion of zero-knowledge against polynomial-time verifiers.

Definition 6.2.7 (SZK-IPCP for languages)**.** We say that $\Gamma = (P, \pi, V)$ is an SZK-IPCP for the language L with SZK $(u(n), \epsilon(n))$ and (adaptive) soundness $1 - \delta(n)$ if

the following holds:

- **Completeness:** If $x \in L$, then $\Pr[\langle P, V^{\pi} \rangle (x) = 1] = 1$.
- **(Adaptive) soundness:** Γ has (adaptive) soundness $1 - \delta$ if for all $x \notin L$, the verifier V is $(1 - \delta(n))$-(adaptively)-sound. Namely for all provers $\widehat{P}$ and (stateful) oracles $\widehat{\pi}$ it holds that $\Pr[\left\langle \widehat{P}, V^{\widehat{\pi}} \right\rangle (x) = 1] \leq \delta(n)$.
- **Statistical zero-knowledge (SZK):** Defined according to the definition of SZK in Definition 6.2.6.

We simply call Γ an SZK-IPCP for L with (adaptive) security u, if Γ is $(1 - 1/u)$-(adaptively)-sound and $(u, 1/u)$-SZK.

Round complexity of IPCPs. By the *round complexity* of an IPCP we mean the number of rounds of interaction between the verifier and the *prover* (and not the PCP oracle) where each round consists of a message from the verifier followed by a message from the prover. In particular, the round i for the verifier starts *after* the $i - 1$'th message of the prover is sent and ends when the i'th query of the verifier to the prover is sent (which will be followed be a message from the prover). Thus in each round the verifier behaves like an oracle-algorithm. The reasons to only consider the interaction with the prover as a factor in round complexity are as follows.

1. In our constructions of SZK-IPCPs, the number of rounds of oracle queries is either equal to one (Theorem 6.3.1) or is bounded by the number of rounds of interaction with the prover (Theorem 6.3.2). On the other hand our negative results (Part 2 of Theorem 6.4.1) is regardless of number of rounds of oracle queries.
2. Regarding the application of IPCPs to the stateless hardware token model, the queries from the PCP oracle are asked *locally* (as opposed to the queries from

the prover which are sent to a remote party). Therefore the number of rounds of interaction with the prover is a more meaningful measure of the efficiency of the system.

Let Γ be an IPCP with j rounds. We can always decompose the verifier's randomness into $r_V = (r_V^1, \ldots, r_V^j)$ so that for the first i rounds only the first i parts $(r_V^1, \ldots, r_V^i)$ are used by the verifier. A trivial decomposition is to let $r_V^1 = r_V$ and r_V^i be the empty string for $i \geq 2$, but that might not be the "natural" decomposition of the randomness.

Definition 6.2.8 (Public-coin IPCPs)**.** By a public-coin IPCP we mean an IPCP where the prover gets to see the coin tosses of the verifier all along the interactions. Namely the verifier's randomness is decomposable into $r_V = (r_V^1, \ldots, r_V^j)$ such that r_V^i is publicly reviled (to the prover) at the beginning of round i and $(r_V^1, \ldots, r_V^i)$ is used by the verifier in round i.

Comments about public-coin IPCPs.

- As a mental experiment we can always assume that the whole randomness r_V is chosen at the beginning and it is only reviled part by part to the prover and is used by the verifier.

- Without loss of generality we can always assume that r_V^i is the i'th message of the verifier to the prover. That is because the prover knows the content of the oracle π and thus can determine the verifier's query at the end of round i by only knowing $(r_V^1, \ldots, r_V^i)$. Despite this fact, it might still be conceptually simpler to describe the verifier's query in other ways.

- Being public-coin is part of the definition of the model and it might change the soundness of an IPCP. Namely, and IPCP might be 0.99-sound when considered

as a (regular) IPCP over an input, but only 0.01-sound when considered as public-coin IPCP.

- When considering adaptive soundness (against stateful oracles), the public-coin IPCP model is only as strong as a two-party (i.e. single prover) interactive protocol. The reason is that the prover and the oracle both get full information about the verifier's messages to both of them and thus can play as a unified party. (For the same reason fully-public-coin multi-prover proof systems are only a special case of single prover proof systems.)

Now we define several forms of composition of IPCPs by extending the corresponding compositions of interactive algorithms (Definition 6.2.2) to the IPCP model. The new component in the IPCP model compared to the model of interactive algorithms is the oracle. The oracle in all the compositions of IPCPs discussed here is simply the combined oracle of the IPCPs being composed. This way the i'th emulated verifier will ask its oracle queries from the i'th oracle by adding a prefix i to its queries and asking them from the combined oracle.

Definition 6.2.9 (Composition of IPCPs)**.** Let $\Gamma_1 = (P_1, \pi_1, V_1), \ldots, \Gamma_k = (P_k, \pi_k, V_k)$ be k IPCPs. Let $\pi = (\pi_1 | \ldots | \pi_k)$ be the oracle combination of $\pi_1, \ldots, \pi_k$, and let U_i be the modified version of V_i which asks its oracle queries from π rather than π_i by adding the prefix i and asking (i, q) from π. We define the sequential $\Gamma_{\mathsf{seq}} = (P_{\mathsf{seq}}, \pi_{\mathsf{seq}}, V_{\mathsf{seq}})$, parallel $\Gamma_{\mathsf{par}} = (P_{\mathsf{par}}, \pi_{\mathsf{par}}, V_{\mathsf{par}})$ and concurrent $\Gamma_{\mathsf{con}} = (P_{\mathsf{con}}, \pi_{\mathsf{con}}, V_{\mathsf{con}})$ composition of $\Gamma_1, \ldots, \Gamma_k$, in order, as the sequential, parallel, and concurrent composition of $(P_1, U_1^{\pi}), \ldots, (P_k, U_k^{\pi})$ (note that $\pi_{\mathsf{seq}} = \pi_{\mathsf{par}} = \pi_{\mathsf{con}} = \pi$). In case of parallel composition the emulated verifiers ask their oracles queries also in parallel. Namely, let U_i^j denote the j'th round of the oracle algorithm U_i (i.e., *after* it asks its $(j-1)$'th query and *before* it asks its j'th query from the prover P_i). Similarly let V_{par}^j denote the

j'th round of the oracle algorithm V_{par}. The oracle algorithm V_{par}^j will be the parallel composition of the oracle algorithms $U_1^j, \ldots, U_k^j$ according to Definition 6.2.4.

The definition above describes the *honest* behavior of the prover, oracle, and the verifier in various forms of composition. The way malicious parties can behave in such compositions is determined both by the definition of the composition and the IPCP model. In particular, although in all forms of composition the emulated V_i asks its oracle queries from π by adding a prefix i (simulating the query being asked from π_i), since the oracle π is accessible by the verifier all along, a malicious verifier is allowed to ask *any* query from the oracle π. Also in the case of concurrent composition, it might be useful for a malicious *stateful* oracle to gather more queries from the verifier before responding to them.

Note that similar to the case of interactive algorithms, the soundness and SZK of the concurrent composition imply those properties (with the same parameters) for the sequential and parallel compositions.

Lemma 6.2.10 (Properties of composition of IPCPs)**.** *Let* $\Gamma_1 = (P_1, \pi_1, V_1), \ldots, \Gamma_k = (P_k, \pi_k, V_k)$ *be* k *IPCPs, and let* $\Gamma_{\mathsf{seq}}, \Gamma_{\mathsf{par}}$ *and* Γ_{con} *be in order their sequential, parallel, and concurrent compositions. It holds that:*

1. **Soundness:** *If* Γ_i *is* $(1-\delta_i)$*-sound for input* x *for all* $i \in [k]$*, then* Γ_{con} *(and in particular* Γ_{seq} *and* Γ_{par}*) will have soundness* $1 - \prod_i \delta_i$ *over the input* x*.*

2. **Adaptive-soundness:** *If* Γ_i *is* $(1-\delta_i)$*-adaptively-sound for input* x *for all* $i \in [k]$*, then* Γ_{seq} *will have adaptive-soundness* $1 - \prod_i \delta_i$ *over the input* x*.*

3. **SZK:** *If* (P_i, π_i, V_i) *is* (u, ϵ_i)*-SZK for the language* L *with a straight-line simulator for all* $i \in [k]$*, then* Γ_{con} *(and in particular* Γ_{seq} *and* Γ_{par}*) will be* $(u, \sum_i \epsilon_i)$*-SZK for the language* L *with a straight-line simulator.*

In the case of sequential composition of interactive protocols it was not necessary for the simulator to be straight-line to get Part 2 of Lemma 6.2.3, but as we will see in the proof of Lemma 6.2.10, in the case of IPCPs since the verifier is allowed to query the oracle whenever she wants, we need the simulator to be straight-line even in the case of sequential composition.

Proof. (of Lemma 6.2.10)

Soundness. Let $\widehat{P}$ be an arbitrary prover and $\widehat{\pi} = (\widehat{\pi}_1|\ldots|\widehat{\pi}_k)$ an arbitrary oracle where $\widehat{\pi}_i$ is the oracle accessed by the emulation of V_i in V_{con}. We claim that $\Pr[\left\langle \widehat{P}, V^{\widehat{\pi}} \right\rangle = 1] \leq \prod_i p_i$ and the reason is as follows. Let $V_i' = V_i^{\widehat{\pi}_i}$ be the interactive algorithm where the oracle $\widehat{\pi}_i$ is hardwired into the algorithm V_i (note that V_i' is inherently inefficient). Now the interactive algorithm $V' = V_{\mathsf{con}}^{\widehat{\pi}}$ is the same as the concurrent composition of $V_1', \ldots, V_k'$. The crucial point is that the interactive algorithm V_i' is well-defined regardless of which other interactive algorithms are run in parallel. That is because π_i is a fixed *oracle* and its answers only depend on the queries that V_i is asking from it (i.e., π_i's answers do not change depending on other queries asked by other V_j's). Let $\delta_i' = \max_P \Pr[\langle P, V_i' \rangle = 1]$, then by the soundness hypothesis it holds that

$$\delta_i' = \max_P \Pr[\langle P, V_i' \rangle = 1] \leq \max_{P,\pi} \Pr[\langle P, V_i^{\pi} \rangle = 1] \leq \delta_i.$$

Therefore by Part 1 of Lemma 6.2.3, it holds that

$$\Pr[\left\langle \widehat{P}, V_{\mathsf{con}}^{\widehat{\pi}} \right\rangle = 1] \leq \max_P \Pr[\langle P, V' \rangle = 1] \leq \prod_i \delta_i' \leq \prod_i \delta_i.$$

Adaptive-soundness. Suppose the sequential executions of $\Gamma_1, \ldots, \Gamma_k$ do not lead to a reject. Then by the $(1 - \delta_i)$-adaptive-soundness of Γ_{i+1}, V_{i+1} will accept with

probability at most δ_i regardless of what has happened in the previous interactions and even if $\widehat{\pi}_{i+1}$ is stateful. Thus the probability that *all* of V_i's accept is bounded by $\prod_i \delta_i$.

SZK. For an arbitrary IPCP $\Gamma = (P, \pi, V)$ let (P^u, V) be the *interactive protocol* in which P^u answers both the queries of V from the prover P and up to u oracle queries of V asked from π. If P, π and P^u are honest, then from the point of view of a verifier V who asks at most u oracle queries there is no difference between interacting with P^u or (P, π). Therefore (P^u, V) is ϵ-SZK for any language L (with a straight-line simulator) if and only if Γ is (u, ϵ)-SZK for L (with a straight-line simulator). Now by Part 2 of Lemma 6.2.3 and using the fact that Γ_i is (u, ϵ_i)-SZK, it follows that (P_i^u, V_i) is ϵ_i-SZK for L. Let $(P_{\mathsf{con}}^u, V_{\mathsf{con}})$ be the concurrent composition of (P_i^u, V_i)'s for $i \in [k]$. By Part 2 of Lemma 6.2.3 $(P_{\mathsf{con}}^u, V_{\mathsf{con}})$ is $(\sum_i \epsilon_i)$-SZK for L. On the other hand, if we restrict a verifier V to ask at most u queries from a combined oracle $\pi = (\pi_1, \ldots, \pi_k)$, the number of queries that V can ask from the i'th sub-oracle π_i (by adding the prefix i to the query) is also bounded by u. Therefore the $(\sum_i \epsilon_i)$-SZK property of $(P_{\mathsf{con}}^u, V_{\mathsf{con}})$ implies that Γ_{con} is $(u, \sum_i \epsilon_i)$-SZK for L.

We shall point out that since in the sequential composition Γ_{seq} a cheating verifier $\widehat{V_{\mathsf{seq}}}$ can ask arbitrary queries to the oracle π_{seq} at any time during the interaction, therefore the behavior of $\widehat{V_{\mathsf{seq}}}$ in Γ_{seq} does *not* necessarily correspond to a cheating verifier interacting in the sequential composition of (P_i^u, V_i)'s. However any cheating verifier $\widehat{V_{\mathsf{con}}}$ participating in Γ_{con} (and in particular participating in Γ_{seq} and Γ_{par}) always corresponds to a cheating verifier interacting in a concurrent composition of (P_i^u, V_i)'s. □

6.2.3 Interactive Locking Schemes

An *Interactive locking scheme* is a commitment scheme implemented in the IPCP model. A similar definition appeared in [108] without the interaction (i.e. only with an oracle), but as we will see in Theorem 6.4.1 *non-interactive* locking schemes are inherently inefficient and therefore not as applicable in cryptographic settings.

Definition 6.2.11 (Interactive locking scheme)**.** Let $\Lambda = (S, \sigma, R)$ be an *efficient* IPCP (where we call S the sender, σ the locking oracle and R the receiver) of the following form:

- The common input is of the form 1^n where n is the security parameter.
- (S, σ) receive a private input $w \in W_n$ which is called the committed message as well as the private randomness r_S. The receiver R gets the randomness r_R.
- The receiver R gets oracle access to the locking oracle σ and R^σ interacts with S in two phases: **(1)** commitment phase and **(2)** decommitment phase. The decommitment phase consists of only one message from the sender S to the receiver R which includes the committed message w and the private randomness r_S used by S. Following this message the receiver R (perhaps after asking more queries from the oracle σ) accepts or rejects.
- Completeness: For any $w \in W_n$ if all parties are honest the receiver accepts with probability one:

$$\Pr[\langle S(w), R^{\sigma(w)} \rangle (1^n) = 1] = 1$$

  where the probability is over the random seeds r_S and r_R.

Then Λ is called an *interactive locking scheme* (ILS) for the message space W_n and if $W = \{0,1\}$, we call Λ a *bit*-ILS. When n is clear from the context we might simply use W rather than W_n to denote the message space.

Definition 6.2.12 (Properties of ILS's)**.** Let $\Lambda = (S, \sigma, R)$ be an ILS. We define the following properties for Λ.

- **Binding:** We define Λ to be $(1-\delta)$-binding if for any sender $\widehat{S}$ and any oracle $\widehat{\sigma}$, with probability at least $1-\delta$ over the interaction of the commitment phase there is at most one possible w such that $\widehat{S}$ can decommit to successfully. More formally, let R' be the interactive algorithm which does the following.

  1. Choose a randomness r_R for the receiver R.
  2. Run the commitment phase of R in Λ.
  3. R' reads all the queries of the oracle $\widehat{\sigma}$ that are of length bounded by the running time of R (this step is inherently inefficient).
  4. Knowing the content of the oracle $\widehat{\sigma}$, R' *accepts* if there exist (w_1, r_1) and (w_1, r_2) such that $w_1 \neq w_2$ and R would accept both of (w_1, r_1) and (w_1, r_2) in the decommitment phase.

  Λ is $(1-\delta)$-binding over the message space W_n iff (S, σ, R') is $(1-\delta)$-sound over the common input $x = 1^n$.

- **Hiding:** Let $\widehat{R}$ be any malicious receiver who asks at most u oracle queries from σ, and let τ_w be the random variable which consists of the transcript of the interaction of R with (S, σ) till the end of the commitment phase when the committed message is $w \in W$. Λ is (u, ϵ)-hiding if for every such malicious receiver $\widehat{R}$ and every $\{w_1, w_2\} \subseteq W$ it holds that $\mathsf{SD}(\tau_{w_1}, \tau_{w_2}) \leq \epsilon$.

- **Equivocability:** Λ is equivocable if there is an efficient sampling algorithm Sam that given (τ, w) where τ is the transcript (including the oracle queries) of the commitment phase of $\left\langle S, \widehat{R}^{\sigma} \right\rangle$ (for an arbitrary receiver $\widehat{R}$) and any $w \in W$, if

$$\Pr[w \text{ is the committed message and } \tau \text{ is the transcript of } \left\langle S, \widehat{R}^{\sigma} \right\rangle] \neq 0$$

  then $\mathsf{Sam}(\tau, w)$ outputs r according to the distribution $(r_S \mid \tau, w)$. Namely r is sampled according to the distribution of the private randomness r_S of (S, σ) conditioned on w being the committed message and τ being the transcript of the commitment phase.

We simply call the ILS Λ u-secure if it is $(1 - 1/u)$-binding and $(u, 1/u)$-hiding.

Public-coin ILS. Since ILS's are IPCPs by definition, we define an ILS to be public-coin similar to the way we defined public-coin IPCPs. Note that the soundness of an ILS might decrease when considered as a public-coin system.

Round-complexity of ILS. Since the interactive part of the decommitment phase of any ILS consists only of one message from the sender S to the receiver R, by the round-complexity of any ILS we only refer to the number of rounds during its commitment phase.

Composition of ILS's. Different forms of composition of IPCPs can be applied to ILS's just as well, but here we are interested in their parallel composition and a specific variant of it. In any k-fold composition of the ILS's the sender receives k private messages $w_1, \ldots, w_k$ where each $w_i \in W$ is used as the private message in the i'th ILS. Therefore the message space of the composed ILS potentially increases

from W to W^k. We use parallel composition to expand the message space. We use a variant of parallel composition which uses the *same* message in all composed ILS's to to amplify the soundness of an ILS.

Definition 6.2.13 (Parallel Compositions of ILS's)**.** Let $\Lambda_1 = (S_1, \sigma_1, R_1), \ldots, \Lambda_k = (S_k, \sigma_k, R_k)$ be k ILS's. By $\Lambda_{\mathsf{par}} = (S_{\mathsf{par}}, \sigma_{\mathsf{par}}, R_{\mathsf{par}})$ we mean the the parallel composition of $\Lambda_1, \ldots, \Lambda_k$ as IPCPs according to Definition 6.2.9. Namely, during the decommitment phase the sender S_{par} sends the message $((w_1, r_{P_1}), \ldots, (w_k, r_{P_k}))$ to the receiver R_{par} who emulates the interactive algorithm R_i over the message (w_i, r_{P_i}) for $i \in [k]$ (and accepts if all of them accept).

By a *same-message* (parallel) composition $\Lambda_{\mathsf{smp}} = (S_{\mathsf{smp}}, \sigma_{\mathsf{smp}}, R_{\mathsf{smp}})$ of $\Lambda_1, \ldots, \Lambda_k$ we mean their parallel composition where the receiver enforces the condition that the same $w_1 = w_2 = \cdots = w_k \in W$ is used as the committed message in all of $\Lambda_1, \ldots, \Lambda_k$. Namely, in the decommitment phase of Λ_{smp}, the sender S_{smp} sends $(w, r_{S_1}, \ldots, r_{S_k})$ to the receiver R_{smp} and the receiver runs the verification algorithm similar to R_{par} but over the message $((w, r_{P_1}), \ldots, (w, r_{P_k}))$.

Note that the message space of the same-message composition Λ_{smp} stays the same as W (as opposed to the regular parallel composition Λ_{par} which expands the message space to W^k).

Lemma 6.2.14 (Properties of parallel compositions of ILS's)**.** *Let $\Lambda_1 = (S_1, \sigma_1, R_1), \ldots, \Lambda_k = (S_k, \sigma_k, R_k)$ be k ILS's with the same round complexity and the same message space W, and let $\Lambda_{\mathsf{par}} = (S_{\mathsf{par}}, \sigma_{\mathsf{par}}, R_{\mathsf{par}})$ and $\Lambda_{\mathsf{smp}} = (S_{\mathsf{smp}}, \sigma_{\mathsf{smp}}, R_{\mathsf{smp}})$ be their parallel and same-message compositions. Then we have:*

1. **Binding:** *If Λ_i is $(1-\delta_i)$-binding for all $i \in [k]$, then Λ_{par} is $(1-\sum_i \delta_i)$-binding and Λ_{smp} is $(1-\prod_i \delta_i)$-binding.*

2. **Hiding:** *If* Λ_i *is* (u, ϵ_i)*-hiding for all* $i \in [k]$*, then* Λ_{par} *and* Λ_{smp} *are both* $(u, \sum_i \epsilon_i)$*-hiding.*

3. **Equivocability:** *If* Λ_i *is equivocable for all* $i \in [k]$*, then both of* Λ_{par} *and* Λ_{smp} *are also equivocable.*

Proof. .

Binding. S_{par} of Λ_{par} gets bound to a message $(w_1, \ldots, w_k)$ if all of S_i's for $i \in [k]$ get bound to a message w_i. By the union bound and the $(1-\delta_i)$-binding of Λ_i with probability at most $\sum_i \delta_i$ by the end of the commitment phase there exist an $i \in [k]$ where S_i is not bound to a fixed w_i. Thus S_{par} has binding at least $(1 - \sum_i \delta_i)$.

On the other hand, since S_{smp} is forced to use the same message w in all of R_i's, S_{smp} gets bound to a message w if for *at least* one $i \in [k]$ R_i gets bound to a message w. Recall that the latter happens whenever the inefficient modification of the receiver R'_i defined in binding part of Definition 6.2.12 rejects and by the binding of Λ_i this rejection happens with probability at least $1 - \delta_i$. Therefore by the soundness amplification of parallel composition of IPCPs (Part 1 of Lemma 6.2.10), with probability $1 - \prod_i \delta_i$ at least one of R'_i's reject in the execution of S_{smp} in which case the sender S_{smp} gets committed to at most one possible message w.

Hiding. We use a standard hybrid argument (which works the same for both Λ_{par} and Λ_{smp}). Fix any two messages $w^1 = (w_1^1, \ldots, w_k^1), w^2 = (w_1^2, \ldots, w_k^2)$ and consider $k+1$ experiments as follows. In the i'th experiment for $0 \leq i \leq k$, we use a parallel composition of $\Lambda_1, \ldots, \Lambda_k$ while for $0 \leq j \leq i$ the committed message in Λ_j is w_j^1 and for $i+1 \leq j \leq k$, the committed message in Λ_j is w_j^2. Fix any malicious receiver $\widehat{R_{\mathsf{par}}}$ who asks at most u queries from its locking oracle σ. Let τ_i be the random variable which consists of the transcript of the commitment phase in the i'th experiment. If $\mathsf{SD}(\tau_0, \tau_k) > \sum_i \epsilon_i$, then there should exist $i \in [k]$ such that $\mathsf{SD}(\tau_{i-1}, \tau_i) > \epsilon_i$. Therefore $\widehat{R_{\mathsf{par}}}$ can distinguish between the experiments $i-1$ and i with advantage ϵ_i. Now we claim that there exists a malicious receiver $\widehat{R}$ that can break the ϵ_i-hiding of Λ_i by ϵ_i-distinguishing the two experiments where in the first one w_i^1 and in the second one w_i^2 is used as the committed message in Λ_i. $\widehat{R}$ will run $\widehat{R_{\mathsf{par}}}$ while

simulating the $k-1$ other senders and oracles of $\widehat{R_{\mathsf{par}}}$ as follows. For $\Lambda_0, \ldots, \Lambda_{i-1}$ $\widehat{R}$ simulates their senders and locking oracles while w_j^1 is used as the committed string in Λ_j and in $k-i$ other games $\Lambda_{i+1}, \ldots, \Lambda_k$ it simulates their senders and locking oracles while w_j^2 is used as the committed string in Λ_j. Then it is easy to see that $\widehat{R}$ will ϵ_i-distinguish between the two experiments because $\widehat{R_{\mathsf{con}}}$ ϵ_i-distinguishes τ_{i-1} from τ_i. But this contradicts the ϵ_i hiding of Λ_i.

Equivocability. Let $\tau_{\mathsf{par}} = (\tau_1, \ldots, \tau_k)$ be the transcript of the interaction of R_{par} with $(S_{\mathsf{par}}, \sigma_{\mathsf{par}})$ where τ_i consists of the part of the transcript for Λ_i and let $w = (w_1, \ldots, w_k)$ be the message that has a non-zero chance of being the private message conditioned on τ_{par} being the transcript. To sample a randomness $(r_{P_1}, r_{P_2}, \ldots, r_{P_k})$ consistent with τ_{par} and w one has to sample r_{P_i} consistent with (τ_i, w_i) for all $i \in [k]$ which is possible by the equivocability of Λ_i's. □

Therefore we can use Λ_{par} to expand the message space at the cost of slight loss in hiding and binding, while Λ_{smp} can be used to amplify the soundness at the cost of slight loss in hiding.

6.3 Statistically Zero-Knowledge IPCP for NP

In this section we show how to construct a $2^{\Omega(n)}$-secure constant-round SZK-IPCP for any language $L \in \mathbf{NP}$ where both the prover and the PCP oracle in our construction can be implemented efficiently given a witness w for $x \in L$. We also show how to achieve a $2^{\Omega(n)}$-*adaptively*-secure SZK-IPCP for any $L \in \mathbf{NP}$ at the cost of poly(n) round-complexity. More formally we prove the following two theorems.

Theorem 6.3.1 (Constant-round SZK-IPCP for **NP**)**.** *For any language $L \in \mathbf{NP}$ there exists a 2-round efficient public-coin SZK-IPCP Γ_{2R} for L with security $2^{\Omega(n)}$.*

Moreover, the simulator of Γ_{2R} *is straight-line and therefore by Lemma 6.2.10 for a small enough constant* c*, a* 2^{cn}*-fold concurrent composition of* Γ_{2R} *remains* $(2^{\Omega(n)}, 2^{-\Omega(n)})$*-SZK.*

Theorem 6.3.2 (Adaptively-secure SZK-IPCP for **NP**)**.** *There exists a (*poly(n)*-round) efficient SZK-IPCP* Γ_{adap} *for* L *with* adaptive*-security* $2^{\Omega(n)}$*.*

The oracle is inherent for SZK. If we only want to achieve computational zero-knowledge, we can remove the PCP oracle from the IPCP model and only the interaction would suffice to achieve this goal (under the computational assumption that one-way functions exist [71]). On the other hand, if the domain of the PCP oracle π were only of size poly(n) (rather than super-polynomial), then a malicious polynomial-time verifier could read all of the information in π and so the language L would be in the class $\mathbf{SZK} \subset \mathbf{AM} \cap \mathbf{coAM}$ which does not contain the class **NP** unless the polynomial hierarchy collapses. This means that the existence of a oracle π with a super-polynomial domain is inherent to achieve unconditional zero-knowledge for **NP**.

Intuition behind Theorem 6.3.1. Our main step to prove Theorems 6.3.1 is to construct an interactive (ILS) (Definition 6.2.11), a primitive corresponding to commitment schemes in the IPCP model. In Theorem 6.4.1 we present an ILS with optimal round complexity (i.e. one round). Then we feed our ILS (as a commitment scheme) into the well-known construction of [71] to achieve zero-knowledge for **NP** with non-negligible soundness. A classical way to amplify the soundness of proof systems (while keeping the round-complexity) in the standard model of interaction is to use parallel composition. We use the fact (Part 1) that parallel composition of IPCP's decreases the soundness error exponentially. The latter result (i.e. Part

1 of Lemma 6.2.14) is interesting on its own since the IPCP model lies in between the single-prover and the multi-prover models and it is known [57] that the parallel repetition does *not* amplify the soundness in a simple exponential form (as one would wish). Secondly, we show that although the parallel composition might hurt the zero-knowledge in general, by crucially using equivocability feature of our ILS (see Definition 6.2.11) one can prove that SZK is preserved under parallel composition.

Lemma 6.3.3 (Followed by Part 1 of Theorem 6.4.1)**.** *There exist an efficient ILS* $\Lambda = (S, \sigma, R)$ *for the message space* $\{1, 2, 3\}$ *with security* $2^{\Omega(n)}$ *which is public-coin and equivocable. Moreover the commitment phase of* Λ *has only one round of interaction.*

To use the construction of [71] we also need the following technical lemma.

Lemma 6.3.4 (Hiding of selective opening of ILS's)**.** *Let* $\Lambda_1 = (S_1, \sigma_1, R_1), \ldots, \Lambda_k = (S_k, \sigma_k, R_k)$ *be ILS's with the same round complexity and the same message space* W *and suppose that* Λ_i *is* (u, ϵ_i)*-hiding for all* $i \in [k]$*. Also let* D *be an arbitrary (perhaps correlated) distribution over* W^k*. Now consider an arbitrary receiver* $\widehat{R}$ *who asks at most* u *oracle queries and interacts with the parallel composition of* $(S_1, \sigma_1), \ldots, (S_k, \sigma_k)$ *in the following experiments* Real *or* Simul*. Then* $\widehat{R}$ *can not distinguish between* Real *and* Simul *with an advantage more than* $3\sum_i \epsilon_i$*. In other words the statistical distance between the view of* $\widehat{R}$ *in the two experiments is at most* $3\sum_i \epsilon_i$*.*

Experiment Real**:**

1. $(w_1, \ldots, w_k)$ *is sampled according to the distribution* D*,* $(r_1, \ldots, r_k)$ *is sampled uniformly at random and* (S_i, σ_i) *receives* (w_i, r_i) *as its private message and randomness.*

2. *$\widehat{R}$ interacts with $(S_1, \sigma_1), \ldots, (S_k, \sigma_k)$ in parallel till the end of the commitment phases.*

3. *$\widehat{R}$ selects an arbitrary set $B \subseteq [k]$ and receives (w_i, r_i) for all $i \in B$.*

4. *$\widehat{R}$ can continue asking up to u oracle queries (including the previous queries asked) from the combined oracle $\sigma = (\sigma_1 | \ldots | \sigma_k)$.*

Experiment Simul**:**

1. *$(r'_1, \ldots, r'_k)$ is sampled uniformly at random and (S_i, σ'_i) receives $(0, r'_i)$ as its private message and randomness.*

2. *$\widehat{R}$ interacts with $(S_1, \sigma'_1), \ldots, (S_k, \sigma'_k)$ in parallel till the end of the commitment phases.*

3. *$\widehat{R}$ selects a subset $B \subseteq [k]$. At this point $(w_1, \ldots, w_k)$ is sampled according to D, and r_i is sampled at random conditioned on being consistent with w_i and the transcript of $\left\langle S_i, \widehat{R}^{\sigma'_i} \right\rangle$. Then $\widehat{R}$ receives (w_i, r_i) for all $i \in B$.*

4. *Now for all $i \in B$ let σ_i be the oracle generated with (w_i, r_i) as its private message and randomness, and for all $i \in [k] \setminus B$ let σ_i be the same as σ'_i. Let the combined oracle be defined as $\sigma = (\sigma_1 | \ldots | \sigma_k)$. $\widehat{R}$ can continue asking up to u oracle queries (including the previous queries) from the combined oracle σ.*

Proof. We will use several hybrid arguments. We first define one more experiment:

Experiment Mixed**:**

1. $(r'_1, \ldots, r'_k)$ is sampled uniformly at random and (S_i, σ'_i) receives $(0, r'_i)$ as its private message and randomness.

2. $\widehat{R}$ interacts with $(S_1, \sigma'_1), \ldots, (S_k, \sigma'_k)$ in parallel till the end of the commitment phases.

3. $\widehat{R}$ selects a subset $B \subseteq [k]$. At this point $(w_1, \ldots, w_k)$ is sampled according to D, and r_i is sampled at random conditioned on being consistent with w_i and the transcript of $\left\langle S_i, \widehat{R}^{\sigma'_i} \right\rangle$. Then $\widehat{R}$ receives (w_i, r_i) for all $i \in B$.

4. Now for *all* $i \in k$ let σ_i be the oracle generated with (w_i, r_i) as its private message and randomness, and let the combined oracle σ be defined as $\sigma = (\sigma_1 | \ldots | \sigma_k)$. $\widehat{R}$ can continue asking up to u oracle queries (including the previous queries) from the combined oracle σ'.

Claim 6.3.5. *For any receiver $\widehat{R}$ who asks at most u oracle queries, the statistical distance between the view of $\widehat{R}$ in* Real *and* Mixed *is at most $\sum_i \epsilon_i$.*

Proof. Let τ_{mixed} and τ_{real} be the transcript of the commitment phases in the experiments Mixed and Real. As a mental experiment suppose:

1. In Mixed, the sampling $(w_1, \ldots, w_m) \leftarrow D_k$ is done at the beginning (similar to Real).

2. In both Real and Mixed, the random seeds $(r_1, \ldots, r_k)$ are chosen after the commitment phases are done, conditioned on the transcript of the commitment phases (τ_{mixed} or τ_{real}) and $(w_1, \ldots, w_m)$.

Now the only difference between Real and Mixed is in the statistical distance between $(w_1, \ldots, w_k, \tau_{\mathsf{mixed}})$ and $(w_1, \ldots, w_k, \tau_{\mathsf{real}})$ in the two experiments. The crucial point is that the distribution of r_i's conditioned on $(w_1, \ldots, w_k, \tau)$ are independent and thus the way they are sampled in Real and Mixed (conditioned on $(w_1, \ldots, w_k, \tau)$) is the same. Now suppose the statistical distance between $(w_1, \ldots, w_k, \tau_{\mathsf{mixed}})$ and

$(w_1, \ldots, w_k, \tau_{\mathsf{real}})$ is more than $\sum_i \epsilon_i$. Then by an average argument we can *fix* the value of $(w_1, \ldots, w_k)$ and still keep the statistical distance to be more than $\sum_i \epsilon_i$. Namely for fixed values of $(w_1, \ldots, w_k)$, the statistical distance between τ_{mixed} and τ_{real} is more than $\sum_i \epsilon_i$. Notice that the difference between Mixed and Real (after fixing $(w_1, \ldots, w_k)$) is that τ_{mixed} is generated with $(0, \ldots, 0)$ as the private messages while τ_{real} is generated with $(w_1, \ldots, w_k)$. But this contradicts the hiding property of parallel composition of ILS's (i.e. Part 2 of Lemma 6.2.14). □

The following claim together with Claim 6.3.5 (and the triangle inequality over the statistical distances) finishes the proof.

Claim 6.3.6. *For any receiver $\widehat{R}$ who asks at most u oracle queries, the statistical distance between the view of $\widehat{R}$ in* Mixed *and* Simul *is at most $\sum_i 2\epsilon_i$.*

Before proving Claim 6.3.6 consider the following two experiments.

Experiment Com0Dec0**:**

1. $\widehat{R}$ interacts with (S_i, σ_i) in the commitment phase conditioned on 0 being the private message of (S_i, σ_i).
2. $\widehat{R}$ can continue asking up to u oracle queries.

Experiment Com0Dec$w$**:**

1. $\widehat{R}$ interacts with (S_i, σ_i) in the commitment phase conditioned on 0 being the private message of (S_i, σ_i).
2. A randomness of the sender r_S is sampled conditioned on the view of $\widehat{R}$ and w being the private message.

3. $\widehat{R}$ can continute asking up to u oracle queries, but the oracle answers will be based on (w, r_S) being the private message and the randomness of the oracle.

Claim 6.3.7. *For any message w, any receiver $\widehat{R}$ who asks at most u oracle queries can distinguish between the* Com0Dec0 *and* Com0Dec$w$ *by an advantage of at most* $2\epsilon_i$.

Proof. Let the experiment Com$w$Dec$w$ be an experiment similar to Com0Dec$w$ where the transcript of the commitment phase was generated conditioned on w being the private message. By the (u, ϵ_i)-hiding of the ILS Λ_i the statistical distance between the view of $\widehat{R}$ in Com0Dec0 and Com$w$Dec$w$ is at most ϵ_i. We can apply Lemma 1.3.2 (with X, Y, and Z being in order the view of $\widehat{R}$ in Com0Dec0, Com$w$Dec$w$, and Com0Dec$w$, and the first part of each random variable being the view of the commitment phase) to conclude that the statistical distance of view of $\widehat{R}$ in Com0Dec0 and Com0Dec$w$ is at most $2\epsilon_i$. □

Proof. (of Claim 6.3.6) Suppose $\widehat{R}$ can distinguish between Simul and Mixed with advantage more than $\sum_i \epsilon_i$. By an average argument we can get fix the sample $(w_1, \ldots, w_k) \leftarrow D_x$ and use it in Simul and Mixed and keep the distinguishing advantage of $\widehat{R}$ more than $\sum_i \epsilon_i$. Now we use a hybrid argument. For $0 \leq j \leq k$, define the j'th hybrid experiment as follows.

Experiment Hyb_j

1. $(r'_1, \ldots, r'_k)$ is sampled uniformly at random and (S_i, σ'_i) receives $(0, r'_i)$ as its private message and randomness.

2. $\widehat{R}$ interacts with $(S_1, \sigma'_1), \ldots, (S_k, \sigma'_k)$ in parallel till the end of the commitment phases.

3. $\widehat{R}$ selects a subset $B \subseteq [k]$. At this point $(w_1, \ldots, w_k)$ is sampled according to D, and r_i is sampled at random conditioned on being consistent with w_i and the transcript of $\left\langle S_i, \widehat{R}^{\sigma'_i} \right\rangle$. Then $\widehat{R}$ receives (w_i, r_i) for all $i \in B$.

4. Now for all $i \in B \cup [j]$ let σ_i be the oracle generated with (w_i, r_i) as its private message and randomness, and for all $i \in [k] \setminus (B \cup [j])$ let σ_i be the same as σ'_i. Let the combined oracle be defined as $\sigma = (\sigma_1 | \ldots | \sigma_k)$. $\widehat{R}$ can continue asking up to u oracle queries (including the previous queries) from the combined oracle σ.

Note that $\mathsf{Hyb}_0 = \mathsf{Simul}$ and $\mathsf{Hyb}_k = \mathsf{Mixed}$ (with the mentioned difference that $(w_1, \ldots, w_k)$ is fixed now). Therefore if $\widehat{R}$ can distinguish between Simul and Real with advantage more than $\sum_i 2\epsilon_i$, then there exists $i \in [k]$ such that $\widehat{R}$ can distinguish between Hyb_{i-1} and Hyb_i with advantage more than $2\epsilon_i$. It is easy to see (similar to the proof of Part 2 of Lemma 6.2.14) that $\widehat{R}$ can be converted into another malicious receiver $\widehat{R}_i$ who asks at most u oracle queries and is able to distinguish between $\mathsf{Com0Dec0}$ and $\mathsf{Com0Dec}w_i$ with advantage more than $2\epsilon_i$. But the latter is not possible because of Claim 6.3.7. □

Therefore Lemma 6.3.4 follows from Claims 6.3.6 and 6.3.5 and the triangle inequality over the statistical distance.[4] □

Now we prove Theorem 6.3.1 using Lemma 6.3.3 and Lemma 6.3.4.

Proof. (of Theorem 6.3.1) Let $L \in \mathbf{NP}$, and let x be an input to an IPCP for L. By using the Cook-Levin reduction one can efficiently compute (from x) a graph G where $x \in L$ iff $G \in$ 3COL, and moreover any witness for $x \in L$ can be efficiently converted

[4]Note that the reductions in various hybrid arguments in the proof of Lemma 6.3.4 are *not* efficient, but they keep the number of queries asked by $\widehat{R}$ bounded (which was sufficient). The reductions can be done efficiently if Λ_i's are equivocable.

into a witness for $G \in$ 3COL and vice versa. So we would assume that the parties always will run the Goldreich-Levin reduction first and then will work with a graph G as their input where G has $n = |N|$ vertices and $m = |M|$ edges. By a padding argument we can always assume that $n \geq |x|$ and so any $u(n)$-secure IPCP for 3COL will yield a $u(n)$-secure IPCP for L as well. Therefore without loss of generality will work with $L =$ 3COL directly.

We use the zero-knowledge construction of [71] which in turn uses a commitment scheme. For the needed commitment scheme we use the ILS $\Lambda = (S, \sigma, R)$ of Lemma 6.3.3.

Construction 6.3.8 (Weakly-sound SZK-IPCP for 3COL)**.** Let $w\colon [n] \to \{1, 2, 3\}$ be a proper 3-coloring for G where $w(i) \neq w(j)$ if $i \neq j$. Let $\Lambda = (S, \sigma, R)$ be the ILS of Lemma 6.3.3 and let w be the secret input given to (S, σ). The IPCP $\Gamma = (P, \pi, V)$ is as follows.

1. The prover P chooses $s \leftarrow S_3$ at random where S_3 is the set of all permutations over the set $\{1, 2, 3\}$. P uses s to "randomize" the coloring w to get a new 3-coloring of G as $c(i) = s(w(i))$ for all $i \in N$. Note that c is also a proper 3-coloring of G. Moreover for any edge $(i, j) \in M$ it holds that $(c(i), c(j))$ is uniformly distributed over $\{1, 2, 3\}^2$ conditioned on $c(i) \neq c(j)$.

2. The prover P and the verifier V will engage in an n-fold parallel composition of the commitment phase of the ILS Λ where $c(i)$ and r_i are, in order, the committed message and the private randomness used by (S_i, σ_i) (in the i'th instance of Λ). The interaction of this step stops right before the decommitment phases start.

3. The verifier V chooses an edge $(i, j) \leftarrow M$ at random and sends (i, j) to the prover P.

4. The prover decommits the i'th and j'th instances of Λ by sending $(c(i), r_i)$ and $(c(j), r_j)$ to the verifier V.

5. The verifier V verifies $(c(i), r_i), (c(j), r_j)$ as decommitments of the i'th and j'th and rejects if either of them fail or if $c(i) = c(j)$.

Claim 6.3.9. *The IPCP $\Gamma = (P, \pi, V)$ of the Construction 6.3.8 has the following properties.*

1. **Completeness:** *If G is a 3-colorable graph, then $\Pr[\langle P, V^{\pi} \rangle (G) = 1] = 1$.*

2. **Soundness:** *Γ is $\frac{1}{2m}$-sound. Namely, if G is not 3-colorable, then $\Pr[\langle P, V^{\pi} \rangle (G) = 0] \geq \frac{1}{2m}$.*

3. **SZK:** *Γ is $(2^{\Omega(n)}, 2^{-\Omega(n)})$-SZK with a straight-line simulator.*

4. **Efficiency:** *Γ can be implemented efficiently (when (P, π) are given a proper coloring w of G as their private input).*

5. **Round-complexity:** *If the commitment phase of Λ has t rounds, then Γ has round complexity $t + 1$.*

6. **Public-coin:** *Γ is public-coin if Λ is public-coin.*

Proof.

Completeness. The completeness follows from the completeness of Λ and the fact that w is a proper coloring.

Soundness. Suppose G is not 3-colorable. Since Γ runs n instances of Λ in parallel, by Part 1 of Lemma 6.2.14, with probability at least $1 - n\delta$, by the end of the commitment phases, for each $i \in [n]$ there is at most one possible $c(i)$ that the prover

can decommit to (together with some randomness r_i) successfully. In the following we assume that such event has happened. Now for each $i \in [n]$ let $c(i)$ be the possible color that the prover can decommit to and let $c(i) = *$ if such color does not exist. Since G is not 3-colorable, with probability at least $1/m$ the verifier chooses an edge (i, j) such that either $c(i) = *$, or $c(j) = *$, or $c(i) \neq c(j)$. But in all these cases the verifier will reject. Therefore the verifier will reject at some point during he interaction with probability at least $(1 - \delta n) \cdot \frac{1}{m}$. Recall that Λ had hiding $1 - \delta \geq 1 - 2^{\Omega(n)}$, which means that Γ has soundness $(1 - n\delta) \cdot \frac{1}{m} \geq \frac{1-n2^{-\Omega(n)}}{m} \geq \frac{1}{2m}$.

SZK. Suppose $\widehat{V}$ is a malicious verifier which asks up to u queries from π where Λ (of Lemma 6.3.3) is (u, ϵ)-hiding. The simulator Sim starts by plugging in a random seed for $\widehat{V}$ and interacting with $\widehat{V}$ similar to the way that (P, π) would do. But the problem is that Sim is not given the coloring w and cannot get the randomized proper coloring c. Instead Sim does the following.

1. Sim uses the trivial (and most probably improper) coloring $c(i) = 0$ for all $i \in N$ and interacts with $\widehat{V}$.

2. After receiving the edge $(i, j) \in M$ from $\widehat{V}$, the simulator chooses $(c(i), c(j)) \leftarrow \{1, 2, 3\}^2$ conditioned on $c(i) \neq c(j)$ and chooses r'_i (resp., r'_j) at random conditioned on being consistent with $c(i)$ (resp., $c(j)$) and the transcript of $\left\langle S_i, \widehat{V}^{\sigma_i} \right\rangle$ (resp., $\left\langle S_j, \widehat{V}^{\sigma_j} \right\rangle$). The latter is possible because of the equivocability of Λ.

3. Sim answers the remaining oracle queries of $\widehat{V}$, but it pretends that $(c(i), r'_i)$ and $(c(j), r'_j)$ are the private message and randomness used in order by σ_i and σ_j. (For queries to other oracles π_t where $t \notin \{i, j\}$, Sim continues using $(0, r_t)$ as the message and the randomness used for the oracle π_t.)

When one is interacting with the honest prover P and the honest oracle π, for any fixed edge $(i,j) \in M$ the distribution of the colors of its vertices $(c(i), c(j))$ is uniformly distributed over $\{1,2,3\}^2$ conditioned on $c(i) \neq c(j)$. Therefore by Lemma 6.3.4, the simulator Sim generates a view for $\widehat{V}$ which is $3\sum_{i\in[n]} \epsilon = (3n\epsilon)$-close to the view of $\widehat{V}$ when interacting with the honest prover and the oracle (P,π). Since Λ is $(u,\epsilon) = (2^{\Omega(n)}, 2^{-\Omega(n)})$-hiding, it follows that Γ has SZK $(2^{\Omega(n)}, 3n2^{-\Omega(n)}) = (2^{\Omega(n)}, 2^{-\Omega(n)})$ with a straight-line simulator.

The efficiency, round-complexity, and public-coin properties are immediate. □

Finally let $\Gamma_{2\mathsf{R}}$ be an mn-fold parallel composition of Γ. By Part 1 of Lemma 6.2.10, $\Gamma_{2\mathsf{R}}$ has soundness $1-(1-1/2m)^{mn} = 1-2^{\Omega(n)}$. Also by Part 3 of Lemma 6.2.10 $\Gamma_{2\mathsf{R}}$ has SZK $(2^{\Omega(n)}, mn2^{-\Omega(n)}) = (2^{\Omega(n)}, 2^{-\Omega(n)})$ with a straight-line simulator. Clearly $\Gamma_{2\mathsf{R}}$ remains complete, efficient, and 2-round and this finishes the proof of Theorem 6.3.1. □

Proof. (of Theorem 6.3.2) To prove Theorem 6.3.2 it is enough to get an SZK-IPCP with non-negligible adaptive soundness. Then by Part 2 of Lemma 6.2.10 we can use the sequential composition to amplify the adaptive-soundness.

Note that if the verifier asks only one query from the oracle, then there is no difference between soundness and adaptive-soundness.

A construction of [15] takes any multi-prover interactive proof system and compiles it into a 2-prover system with non-negligible soundness where the verifier asks only one query from the oracle. The construction of [15] is as follows. The verifier sends its randomness r_V to the first prover and gets a full simulation of its interaction with the original multi-prover system. Then, in order to make sure that the first prover did not lie, one of the query/answer pairs claimed by the first prover is chosen at random and is verified by the second prover. If the answers were different the verifier

rejects. It is easy to see that if the first prover does not simulate the game honestly as a simulation of the multi-prover case, then it will be caught with probability roughly $1/v$ where v is the total number of queries of the interaction being simulated. (We refer the reader to [15] for more details).

As a first try, we use the construction of [15] to compile our SZK-IPCP of Theorem 6.3.1 into one with non-negligible adaptive-soundness and only one oracle query. Namely, we starts ask the verifier to send its randomness to the prover who simulates the whole execution of the original system. Then we choose one of the query/answer pairs claimed by the prover and verify its correctness with the oracle (who is supposed to know the answer to such queries). Unfortunately, the problem with this construction is that the verifier is also able to "reset" the original prover through the new oracle. That is because the new oracle is supposed to know the answer to all the queries to the original prover and there is no restriction on which queries the verifier will ask from the oracle. Therefore we loose the SZK property.

To keep the SZK property, as a second try, we change the construction above as follows. The verifier will choose only one of the *oracle* queries, simulated by the new prover, and verifies that query with the new oracle. Now if the prover lies about any of the oracle-type simulated queries, it still will be caught by the verifier with non-negligible probability. Unfortunately, this time we might loose the soundness. The reason is that if the soundness of the original system relies on the oracle queries being hidden from the prover, since in the new simulation we are revealing the oracle queries of the original system to the prover, the prover might honestly simulate the oracle queries but use this knowledge to lie about the simulated queries of the original prover. But the good news is that if the original construction of SZK-IPCP was public-coin, then the soundness is still guaranteed even if the prover gets to see the oracle queries, and so in this case we do *not* loose the soundness! More formally we use the following

construction.

Construction 6.3.10 (From soundness to weak adaptive-soundness)**.** Let $\Gamma = (P, \pi, V)$ be an IPCP. We construct new IPCP $\Gamma' = (P', \pi', V')$ as follows.

- **PCP oracle:** The (honest) oracle π' will be the same as π.
- **The interaction of P' and V':**
  1. V' chooses a randomness r_V for V and interacts with P' similar to the interaction of V and P. The main difference is that any time that V needs to ask a query from π, V' will ask this query from the *prover* P' instead.
  2. At this time V' has a full description of a possible interaction between V^{π} and P and rejects if V would reject in this interaction.
  3. Suppose $q_1, \ldots, q_v$ are the oracle queries that were asked from the prover P' and let a_i be the answer claimed by P' for the oracle query q_i. If V' didn't reject in the previous step, it will chooses a random $i \in [v]$ and queries q_i from the oracle π'. Let $a'_i = \pi'(q_i)$ be the answer. V' rejects if $a_i \neq a'_i$.

Claim 6.3.11 (Properties of Construction 6.3.10)**.** *Let $\Gamma = (P, \pi, V)$ be an (efficient)* public-coin *IPCP for the language L which is $(1-\delta)$-sound, (u, ϵ)-SZK and public-coin where the honest verifier V asks at most v oracle queries. When Γ is used in Construction 6.3.10 the result is an (efficient) IPCP $\Gamma' = (P', \pi', V')$ for the language L which is $(u - v, \epsilon)$-SZK, and is $(1-\delta)/v$-*adaptively-*sound.*

Proof. .

SZK. Any simulator Sim for Γ with parameters (u, ϵ) can be used to get a simulator Sim' for Γ' with parameters $(u - v, \epsilon)$. The reason is that the view of any malicious

verifier $\widehat{V'}$ in Γ' who asks at most $u-v$ can be simulated by another malicious verifier $\widehat{V}$ in Γ who uses $\widehat{V'}$ as a black box (and in a straight-line manner). The verifier $\widehat{V}$ runs $\widehat{V'}$ and whenever $\widehat{V'}$ asks an *oracle* query from the prover P', $\widehat{V}$ also asks this query from the oracle π. Thus any (straight-line) simulator for $\widehat{V}$ can be used to simulate the view of $\widehat{V'}$ with the same statistical distance ϵ.

Adaptive soundness. Since the verifier asks only one query from π', it does not matter whether π' is stateful or not. More formally, for any potentially stateful π' we define a (stateless) oracle π'' which given the query x answers according to π' when π' is asked *only* the query x. Note that from the point of view of V' there is no difference between π' and π'', so we might as well assume that V' asks its query from the *stateless* oracle π''.

Let Rej be the event that V' rejects. By the soundness of Γ as a *public-coin* IPCP if P' answers all of $q_1, \ldots, q_v$ honestly according to π'', then it would hold that $\Pr[\mathsf{Rej}] \geq 1-\delta$. It is crucial that Γ is public-coin, because in the interaction of V' and P', P' gets to know the oracle queries of V and the soundness should hold even in this case. But a malicious prover $\widehat{P'}$ can change $\Pr[\mathsf{Rej}]$ by lying about the answer to some of q_i's. Let Lie be the event that there exists $i \in [v]$ such that $a_i \neq \pi''(q_i)$, and let NoLie be the complement event that $a_i = \pi''(q_i)$ for all $i \in [v]$. By the $(1-\delta)$-soundness of the public-coin IPCP Γ it still holds that:

$$\Pr[\mathsf{Rej} \wedge \mathsf{NoLie}] \geq (1-\delta) - \Pr[\mathsf{Lie}]. \tag{6.3.1}$$

On the other hand if Lie happens then P' will be caught with probability at least $1/v$, and so

$$\Pr[\mathsf{Rej} \mid \mathsf{Lie}] \geq 1/v. \tag{6.3.2}$$

Therefore we conclude that

$$\begin{aligned}
\Pr[\mathsf{Rej}] &= \Pr[\mathsf{Rej} \wedge \mathsf{NoLie}] + \Pr[\mathsf{Rej} \wedge \mathsf{Lie}] \\
&\geq \max(0, 1 - \delta - \Pr[\mathsf{Lie}]) + \Pr[\mathsf{Lie}] \cdot \Pr[\mathsf{Rej} \mid \mathsf{Lie}] && \text{by Inequality 6.3.1} \\
&\geq \max(0, 1 - \delta - \Pr[\mathsf{Lie}]) + \Pr[\mathsf{Lie}] \cdot \frac{1}{v} && \text{by Inequality 6.3.2} \\
&\geq (1 - \delta)/v. && \text{Achieved by } \Pr[\mathsf{Lie}] = 1 - \delta
\end{aligned}$$

□

We use Construction 6.3.10 over the IPCP $\Gamma = (P, \pi, V)$ of Theorem 6.3.1 to get a new IPCP $\Gamma' = (P', \pi', V')$. Let $v = \text{poly}(n)$ be the number of oracle queries of V in Γ. Since Γ is public-coin and is $2^{\Omega(n)}$-secure, therefore Γ' will have adaptive-soundness $1/(2mv) = 1/\text{poly}(n)$ and SZK $(2^{\Omega(n)} - v, 2^{-\Omega(n)}) = (2^{\Omega(n)}, 2^{-\Omega(n)})$.

Finally let Γ_{adap} be a mvn-fold sequential composition of Γ'. By Part 2 of Lemma 6.2.10, Γ_{adap} has adaptive-soundness $1 - (1 - 1/(2mv))^{mvn} = 1 - 2^{\Omega(n)}$. Also by Part 3 of Lemma 6.2.10 Γ_{adap} has SZK $(2^{\Omega(n)}, mvn2^{-\Omega(n)}) = (2^{\Omega(n)}, 2^{-\Omega(n)})$ and this finishes the proof of Theorem 6.3.2. □

6.4 Interactive Locking Schemes

In this section we study ILS's and will construct an ILS with optimal round complexity.

Theorem 6.4.1. *(A round-optimal ILS) Let $\ell(n) = \text{poly}(n)$, then*

1. *There exist an efficient ILS $\Lambda = (S, \sigma, R)$ for the message space $\{0,1\}^\ell$ with security $2^{\Omega(n)}$ which has a commitment phase of only one round and is public-coin.*

2. *Any ILS with a noninteractive commitment phase needs an inefficient oracle σ and thus Λ has optimal round-complexity (as an efficient ILS).*

First, in Section 6.4.1 we present a general construction for ILS's from any interactive hashing scheme (IHS) (see Definition 6.4.2) where the round-complexity of the used IHS equals the round-complexity of the commitment phase of the constructed ILS. Using known constructions for IHS we get an ILS with a 2-round commitment phase. We show that this is the best one can get through this approach by showing that any IHS with the parameters needed for our construction requires at least 2 rounds (see Proposition 6.4.5).

Then in Section 6.4.2 we present a direct construction for ILS with only one round of interaction in the commitment phase. Both constructions of Sections 6.4.1 and 6.4.2 are $2^{\Omega(n)}$-secure and public-coin. Thus the 1-round construction of Section 6.4.2 proves Part 1 of Theorem 6.4.1.

Finally in Section 6.4.3 we will show that at least one round of interaction is needed in the commitment phase of any *efficient* ILS (proving Part 2 of Theorem 6.4.1).

6.4.1 ILS from IHS

Interactive hashing schemes (IHS — Definition 6.4.2) and their variants have been of great use in designing cryptographic protocols. Although the core central properties needed for IHS's in all these applications is of an information theoretic flavor, in the applications of IHS's in the computational regime [83, 85, 132, 133], compared to the applications in the information theoretic regime [36, 36, 48]) some extra properties are required for the IHS used. Here we use a minimal definition whose properties hold in all existing forms of IHS in the literature and show that the existence of any IHS according to this definition implies the existence of ILS.

Definition 6.4.2 (Interactive hashing)**.** An *interactive hashing scheme* (IHS) $\Psi = (S, R)$ is a two party protocol between a sender S and a receiver R where both are given 1^n as the security parameter and S is also given $w \in M$ as the private message. By the end of the protocol the transcript of the protocol determines two outputs $\{w_0 \neq w_1\} \subseteq \{0,1\}^n$ for which the following properties hold.

- **Completeness:** If S and R are honest, then $w \in \{w_0, w_1\}$.
- **Binding:** The protocol is ρ-binding, if for every *fixed* set $T \subset W$ of size $|T| \leq \rho \cdot |W|$ and any malicious sender $\widehat{S}$ the probability that both of w_0 and w_1 lie in T is at most $1/2$; namely $\Pr_{\langle \widehat{S}, R\rangle = (w_0, w_1)}[\{w_0, w_1\} \subset T] \leq 1/2$.
- **Hiding:** If the sender S is honest, then for any transcript τ of the protocol which determines the outputs (w_0, w_1) it holds that $\Pr_{w \leftarrow \{w_0, w_1\}}[w = w_0 \mid \tau] = \Pr_{w \leftarrow \{w_0, w_1\}}[w = w_1 \mid \tau] = 1/2$. Therefore if, $w \leftarrow W$ is chosen uniformly at random, w_0 and w_1 will have the same chance of being equal to w.

We call Ψ *efficient* if both S and R are efficient and *public-coin* if the messages of the receiver R only consist of her public coin tosses.

It is easy to see that if R simply sends a 2-to-1 pairwise-independent hash function $h\colon \{0,1\}^n \to \{0,1\}^{n-1}$ to S and S sends back $h(w)$, it gives a ρ-binding IHS for $\rho \approx 2^{-n/2}$. But, as we will see in Lemma 6.4.3 for our construction of ILS from IHS (Lemma 6.4.3) we need IHS's with non-negligible binding $\rho = \Omega(1/\operatorname{poly}(n))$.

Lemma 6.4.3 (ILS from IHS)**.** *Let $\Psi = (S_\Psi, R_\Psi)$ be a k-round IHS with binding ρ over the message space $\{0,1\}^n$ with a deterministic sender S_Ψ. Then there exists a bit-ILS $\Lambda = (S, \sigma, R)$ with a k-round commitment phase which is $1/2$-binding and $(2^{n/2}, 2^{-n/2})$-hiding. Moreover if Ψ is efficient, then (S, σ) are efficient and the receiver R runs in time $\operatorname{poly}(n, 1/\rho)$ (and so if in addition $\rho = 1/\operatorname{poly}(n)$, then Λ is efficient as well).*

[133] presented an IHS with binding $\rho = \Omega(1)$ at the cost of $\text{poly}(n)$ rounds of interaction. [48] showed how to achieve a constant-round IHS with $1/\text{poly}(n)$ binding. In all the known constructions of IHS the sender is deterministic, and in our construction of ILS from IHS (Lemma 6.4.3) we use this property. The determinism of the sender is not crucial to us and if the sender is randomized, then we only need the following sampling property to hold: Given any transcript τ of the protocol determining (w_0, w_1) and any $i \in \{0,1\}$ one can efficiently sample r_i from the sender's space of randomness according to $r_i \leftarrow (r_S \mid \tau, w = w_i)$.

By using any known k-round efficient IHS with $1/\text{poly}(n)$-hiding and Lemma 6.4.3, we get an efficient ILS Λ which is $1/2$-binding and $(2^{\Omega(n)}, 2^{-\Omega(n)})$-hiding. Now Part 1 of Lemma 6.2.14 yields that an n-fold parallel composition of Λ is $2^{\Omega(n)}$-secure and finally an ℓ-fold same-message parallel composition expands the message space to $\{0,1\}^\ell$ while we keep the $2^{\Omega(n)}$-security. Therefore by using the 2-round IHS scheme of [48] and Lemma 6.4.3 we derive Part 1 of Theorem 6.4.1 (but still without the optimal round-complexity).

Intuition. Our approach in the proof of Lemma 6.4.3 is to use the well-known reduction from statistically hiding bit commitments to IHS in the *computational* setting [125] and its black-box proof of security. Roughly speaking the idea of [125] is to run the IHS over the message $y \in \{0,1\}^n$ where $y = f(x)$ and f is a permutation. Now if f is a one-way permutation the fraction of points y which the malicious sender $\widehat{S}$ is able to invert is a negligible $\epsilon = \text{neg}(n)$ fraction of $\{0,1\}^n$. Therefore if the ILS is ρ-binding for non-negligible ρ (and in particular $\epsilon < \rho$) with a noticeable probability, by the end of the IHS with outputs $\{y_0, y_1\}$, there is at most one y where $\widehat{S}$ can invert (i.e., present x such that $y = f(x)$) and claim that y was his private message. So if at the last step of the commitment phase $\widehat{S}$ sends a bit $d = b \oplus c$ where b is his

bit-message and y is such that $y = y_c$, then she gets committed to only one possible value for b for which she can find a consistent c.[5]

But here we are interested in unconditionally secure protocols and we can not rely on the existence of one-way permutations or any other computational assumption. The idea is to use the locking oracle σ to hold a random (non-zero) point function $\sigma(x) = y$ where the sender knows the image y and the preimage x and thus can "invert" y to x. By using the IHS over the image y, the sender gets committed to a unique y (with probability at least $1/2$) assuming that there are fewer than $\rho \cdot 2^n$ many "invertible" points in the set $\sigma(\{0,1\}^n)$. But the receiver can guarantee that the latter is the case by picking $100/\rho$ random samples $x' \leftarrow \{0,1\}^n$ (conditioned on $x \neq x'$) and verifying that $\sigma(x') = 0$.

Proof. (of Lemma 6.4.3) We first describe the bit-ILS $\Lambda = (S, \sigma, R)$.

Construction 6.4.4 (Bit-ILS from IHS)**.** Let $b \in \{0,1\}$ be the private message hold by (S, σ). We will use the IHS Ψ with the properties described in Lemma 6.4.3 over the message space $\{0,1\}^n$ with binding ρ.

The commitment phase of Λ:

1. The common randomness r_R of (R, σ) will be $(x, y) \leftarrow \{0,1\}^n \times \{0,1\}^n$ where $y \neq 0$. The locking oracle σ will contain a point function: $\sigma(x) = y$ and $\sigma(x') = 0$ for $x' \neq x$.

2. S and R engage in the IHS Ψ where the sender S uses y as the private message of S_Ψ.

[5]The argument needs to be formalized through an *efficient* reduction and we refer the reader to the rather complicated full proof of [125].

3. Let b be the private bit given to S, and let $(y_0 < y_1)$ be the output of the protocol Ψ and let c be such that $y = y_c$. As the last message of the commitment phase the sender S sends the bit $d = b \oplus c$ to the receiver R. Note that this does not require an extra round of interaction since d can be concatenated to the last message of the sender.

The decommitment phase of Λ:

1. S sends b and (x, y) to R.

2. R verifies that $\sigma(x) = y$ and $y = y_{d \oplus b}$ (and rejects otherwise).

3. For $i \in [n/\rho]$ the receiver R samples $x'_i \leftarrow \{0,1\}^n$ conditioned on $x'_i \neq x$ at random and verifies that $\sigma(x'_i) = 0$ holds for all $i \in [n/\rho]$ (and rejects otherwise).

Now we prove the properties of the ILS Λ of Construction 6.4.4.

Completeness is immediate.

Binding. Let $T = \{\sigma(x) \mid x \in \{0,1\}^n\} \setminus \{0\}$ and $T^{-1} = \{x \mid \sigma(x) \neq 0\}$ be the set of non-zero points in the oracle σ. There are two cases: either T^{-1} has size $|T^{-1}| \geq \rho \cdot 2^n$ or $|T^{-1}| < \rho \cdot 2^n$. If the first case $|T^{-1}| \geq \rho \cdot 2^n$ holds, then with probability at least $1-(1-\rho)^{n/\rho} > 1-2^n$ one of the receiver's samples $\{x'_i\}$ will be sampled from T^{-1} and then the receiver R rejects. (The way we described the protocol the random samples x'_i's are chosen in the decommitment phase, but they are part of the randomness r_R of the receiver and are chosen in the beginning of the commitment phase.)

On the other hand, if the second case $|T^{-1}| < \rho \cdot 2^n$ holds, it implies that the size of $|T| \leq |T^{-1}| < \rho \cdot 2^n$ is small as well. Now by the binding property of the IHS Ψ, with probability at least $1/2$ one of $\{y_0, y_1\}$ lies out of T in which case there is

only one way for the sender S to decommit to a bit b consistent with y_0, y_1 and d. Therefore the 1/2-binding holds in either of the cases.

Hiding. If a malicious receiver $\widehat{R}$ asks up to $2^{n/2}$ queries from the oracle σ, they will be all answered zero with probability at least $1 - 2^{-n/2}$ by the uniformity of $x \leftarrow \{0,1\}^n$. But in that case the oracle queries of $\widehat{R}$ do not reveal any information about y and the hiding of the ILS Γ follows from the hiding of the IHS Ψ.

Equivocability. Let τ be the transcript of any malicious receiver $\widehat{R}$ by the end of the commitment phase. Given (b, τ) the sampler algorithm Sam does the following. Let $\sigma(x'_1) = y'_1, \ldots, \sigma(x'_t) = y'_t$ be the oracle queries asked by $\widehat{R}$ from σ appeared in the transcript τ and let (y_0, y_1) be the output of the executed IHS. If there exist $i \in [t]$ such that $y'_t \neq 0$, then it means that $\widehat{R}$ has found the committed bit $b = d \oplus c$ where c is such that $y'_i = y_c$. In this case the sampler Sam simply outputs (x'_i, y'_i). On the other hand if $y'_i = 0$ for all $i \in [t]$, then the sampler chooses $y = y_c$ where $c = b \oplus d$ and chooses $x \leftarrow \{0,1\}^n \setminus \{x'_1, \ldots, x'_t\}$ and outputs (x, y) (pretending that $\sigma(x) = y$). □

By using the IHS of [48] (which has the fewest number of rounds among known IHS's) we get a 2-round ILS. The following proposition shows that the Construction 6.4.4 of ILS's from IHS's is not able to achieve efficient ILS's with one round (in the commitment phase).

Proposition 6.4.5 (Nontrivial IHS's need two rounds)**.** *Any IHS Ψ with at most 3 messages exchanged between the sender and the receiver has binding at most $\rho = 100/\sqrt{|W|}$. Therefore for $|W| = 2^n$, any IHS Ψ with non-negligible binding ρ needs at least two rounds of interaction.*

Proof. For starters we assume that Ψ has only one round (i.e. two messages) where the receiver sends its message first. Later we will extend the result to 3-message protocols. Suppose the first message sent from the receiver is q and is followed by an answer a from the sender. In the following discussion we fix a query q. Now any answer a corresponds to two elements $\{w_1^a, w_2^a\} \subset W$ which determines an edge in a graph G_q with the vertex set W. Each edge a has a probability of being used as the sender's message. By duplicating the edges we can change G_q into a multi-graph such that the distribution of the message of the sender corresponds to selecting a random edge from G_q. The hiding property implies that any two connected vertices x and y have the same degree $d(x) = d(y)$. Therefore each connected component of G_q is a regular graph. Now we show that such graphs have a relatively large matching.

Lemma 6.4.6 (Large matchings in regular graphs)**.** *Let G be any graph with v vertices such that the vertices have degree at least one and each connected component of G is regular. Then G has a matching of size at least $v/4$.*

Proof. By the Vising theorem each connected component of G can be properly colored with $\Delta + 1$ colors where Δ is the degree of the vertices of that component. Since each color forms a matching, therefore each component of G with u vertices has a matching of size at least $\frac{u\Delta}{2(\Delta+1)} \geq u/4$. Therefore G has a matching of size at least $v/2$. □

Let M be the matching of size $|W|/4$ in G guaranteed by Lemma 6.4.6. The sender samples a random set $T \subset W$ by choosing each $w \in W$ with probability $10/\sqrt{|W|}$. Each edge of the matching M gets covered by T with probability at least $100/|W|$. Therefore with probability at least $1-(1-100/|W|)^{|W|/4} > 1-2^{-25} > 99/10$ at least on of the edges in M gets covered.

Now we let q, the message of the receiver, also be chosen according to its distribution. For each such q, when we construct the graph G_q, the random set T covers one of the edges of G_q with probability at least 99/100. Therefore, with probability at least 9/10 over the choice of T, the set T has the property that: with probability at least 9/10 over the choice of q, at least one of the edges of G_q is covered by T. We call such T a good set.

On the other hand by Markov inequality with probability at least 9/10, the size of T is bounded by $100\sqrt{|W|}$. Therefore by a union bound with probability at least 8/10, T is both a good set and also of size at most $100\sqrt{|W|}$. This set T can be used by the sender to break the binding of Ψ, because with probability at least 9/10 over the first message q of the receiver, the sender is able to send a message a (corresponding to the edge covered by T) which determines two elements in the set T as the output of the protocol.

Now we study 3-message protocols. If we add one more message q_1 from the receiver to to the sender (in addition to the the 2-messages in the protocols studied above), the last message of the receiver does not restrict the sender's cheating power, because the set of messages that the sender can claim to be his message does not depend on the last message of the sender.

Now suppose the protocol has three messages where the sender starts the interaction. In this case, the first message of the sender a_0 might decrease the space of possible values for w which are consistent with a_0. Now the sender *fixes* a value a_0 for his first message. Let W' be the set of values for w which are consistent with a_0. By using the same attack above (for the protocols which start with a_0), the sender can still find a set T of size at most $100\sqrt{|W'|} < 100\sqrt{|W|}$ to break the binding of Ψ.

□

6.4.2 A 1-round ILS

Theorem 6.4.7 (A 1-round ILS)**.** *There exists an efficient bit-ILS Λ_{1R} which is $2^{n/10}$-secure and has only one round of interaction in the commitment phase.*

Before proving Theorem 6.4.7 we need to develop some basis tools.

Thus we also get the following lemma, whose proof is immediate.

Lemma 6.4.8. *For $n > m$ let $\mathcal{A}$ be the family of $n \times m$ Boolean matrices as follows. To get a uniform member of A, choose the first $n - m$ rows all at random, and take the last m rows to be an independently chosen random member of* full rank m*. Then for any $0 \neq x \in \{0,1\}^n$, it holds that $\Pr_{A \leftarrow \mathcal{A}}[xA = 0] \leq 2^{-m}$ (and equivalently for any $x_1 \neq x_2 \in \{0,1\}^n$ and $y \in \{0,1\}^m$, it holds that $\Pr_{A \leftarrow \mathcal{A}}[x_1 A = x_2 A] \leq 2^{-m}$).*

Proof. (of Theorem 6.4.7) We use the following construction for the ILS Λ_{1R}.

Construction 6.4.9 (A 1-round ILS)**.** Suppose $b \in \{0,1\}$ is the private message given to sender and the oracle (S, σ), and suppose R is the receiver. Let $m = 3n/4$. Below we associate $\{0,1\}^n$ with the integers $[0, 2^n)$ and all additions and subtractions below are modulo 2^n.

The commitment phase of Λ_{1R}:

1. Sender S chooses $a \leftarrow \{0,1\}^n$ at random. Let f_b be the function: $f_b(x) = 1$ iff $a \leq x < a + 2^m$, and let f_{1-b} be the zero function over $\{0,1\}^n$. The locking oracle will be the combination of the two functions $\sigma = (f_0 | f_1)$ (indexed by the first bit of the query to σ).

2. Receiver R samples $A \leftarrow \mathcal{A}$ from the family of matrices of Lemma 6.4.8 and sends A to S.

3. Sender S checks that the last m rows of A are independent, and if so he sends $h = aA$ to the receiver R.

The decommitment phase of Λ_{1R}:

1. Sender S sends (b, a) to the receiver R.
2. Receiver R does the following checks and rejects if any of them does not hold.
   (a) Check that $aA = h$.
   (b) Check that $f_{1-b}(a) = 0$, and $f_b(a) = 1$.
   (c) For each $i \in [0, m]$, sample $10n$ random points from $[a, a + 2^i)$ and check that $f_b(x) = 1$ for all of them, and also sample $10n$ random points from $(a - 2^i, a - 1]$ and check that $f_b(x) = 0$ for all of them

Now we study the properties of the ILS Λ_{1R}.

Completeness is immediate.

Binding. As a mental experiment we pretend that the randomness used during the decommitment phase by R is chosen in the decommitment phase (rather than in the beginning of the commitment phase).

For a fixed locking oracle σ, Let X_0 (resp., X_1) be the set of possible values of a that sender S can send to the receiver R as the decommitment of $b = 0$ (resp., $b = 1$) and get accepted in the decommitment phase with probability at least 2^{-2n}. We prove that by the end of the commitment phase, with probability at least $1 - 2^{-n/8}$, it holds that $|X_0| = 0$ or $|X_1| = 0$ which means that the sender has only one way to decommit the value b and get accepted with probability more than 2^{-2n}. But now if we choose the receiver's randomness in the commitment phase, since there are at

most 2^{n+1} possible values for (b, a), it follows by a simple average argument that with probability at least $1-2^{2n-n-1}$ over the commitment phase, the prover gets committed to only one possible value for (b, a) which he can use to pass the decommitment phase successfully.

Claim 6.4.10. *$X_0 \cap X_1 = \varnothing$.*

Proof. If $a \in X_0 \cap X_1$. Then when a is used as the decommitment of 0, in Step 2b of the decommitment phase the receiver R checks that $f_0(a) = 1, f_1(a) = 0$. On the other hand in the case of decommitting to 1, receiver R checks that $f_b(a) = 0, f_{1-b}(a) = 1$, but they can't both hold at the same time. □

Claim 6.4.11. *It holds that $|X_0| \leq 2^{n-m}$ and $|X_1| \leq 2^{n-m}$.*

Proof. We show that if $\{a, a'\} \subset X_0$ then $|a - a'| \geq 2^m$ (and this would show that $X_0 \leq 2^n/2^m$). Assume on the contrary that $a' < a$ and $a - a' < 2^m$. Let $i \in [1, m]$ be such that $2^{i-1} \leq a - a' < 2^i$. Then by the pigeonhole principle ether at least half of $\sigma([a', a])$ are zero or at least half of the values $\sigma([a', a])$ are one. Without loss of generality let assume that at least half of $\sigma([a', a])$ is zero. In this case at least $1/4$ of the values $\sigma([[a', a' + 2^i)])$ are zero. But then by Step 2c of the decommitment phase $(0, a')$ will be accepted with probability at most $(3/4)^{10n} < 2^{-2n}$, and therefore $a' \notin X_0$ which is a contradiction. □

Claim 6.4.12. *With probability at least $1 - 2^{\Omega(n)}$ over the choice of A, it holds that $|X_0| = 0$ or $|X_1| = 0$.*

Proof. Fix any pair $a_0 \in X_0$ and $a_1 \in X_1$, we know that $a_0 \neq a_1$. Therefore, $\Pr_A[a_0 A = a_1 A] = \Pr_A[(a_0 - a_1)A = 0] \leq 2^{-m}$. Claim 6.4.11 yields that there are at most $2^{n-m}2^{n-m}$ such pairs, so by using a union bound, with probability at least $1 - 2^{-m}2^{2n-2m} = 1 - 2^{2n-3m}$ over the choice of A, it holds that $X_0 A \cap X_1 A = \varnothing$ which

implies that if the sender sends any hash value h, the consistency check of Step 2a of the decommitment phase either makes $|X_0| = 0$ or $|X_1| = 0$. $\square$

As we said before Claim 6.4.12 implies that with probability $1 - \text{poly}(n) \cdot 2^{2n-3m} = 1 - \text{poly}(n) \cdot 2^{-n/4} \geq 1 - 2^{-n/8}$ over the interaction in the commitment phase the sender gets bound to a fixed $b \in \{0, 1\}$ to which he can decommit successfully.

Hiding. Suppose receiver R can ask at most $u \leq 2^{n/8}$ queries from the locking oracle σ. We claim that before sending the matrix A, all of receiver R's queries to σ are answered zero with probability at least $1 - 2^{-n/4}$. To see why, think of Z_{2^n} as being divided into $2^{n-m} = 2^{n/4}$ equal intervals such that a is the beginning of one of them. Since receiver R asks up to $2^{n/8}$ queries, before sending the matrix Z, he will ask a query from the interval beginning with a with probability at most $2^{n/8}/2^{n/4} = 2^{-n/8}$. Therefore (up to $2^{-n/8}$ statistical distance in the experiment) we can assume that the matrix A is chosen by receiver R independently of a.

After receiving h, the information that the receiver R knows about a is that it satisfies the equation $aA = h$. If we choose and fix the first $n - m$ bits of (a potential) a, then the remaining bits are determined uniquely because the last m rows of A are full rank. It means that for every $y \in [0, 2^{n-m})$ there is a unique solution for a in the interval $[y2^m, y2^m + 2^m)$, and they are all equally probable to be the true answer from the receiver's point of view.

Now again we claim that (although there are 2^m nonzero points in f_b) all the queries that the receiver R asks from f_b are answered 0 with probability at least $1 - 2^{-n/8}$. Let $Z = \{z \mid zA = h\}$ be the set of possible values for a. For $z \in Z$, let $I(z) = [z, z + 2^m)$. We claim that no $x \in \{0, 1\}^n$ can be in $I(z)$ for three different z's from Z. To see why, let $z_1 < z_2 < z_3$ and that $x \in I(z_1) \cap I(z_2) \cap I(z_3)$. But now the interval $[y2^m, y2^m + 2^m)$, containing z_2 separates z_1 and z_3, and so $z_3 - z_1 > 2^m$.

Therefore $I(z_1) \cap I(z_3) = \varnothing$ which is a contradiction. So, if the receiver R asks u queries from f_b, he can ask queries from $I(z)$'s for at most $2u$ different z's (out of 2^{n-m} many of them). As a mental experiment assume that a is chosen from Z after the receiver R asked his queries, it holds that $I(a)$ will be an interval that the receiver R never asked any query from with probability at least $1 - u/2^{n-m} \geq 1 - \cdot 2^{-n/8}$. Therefore with probability at least $1 - 2^{-n/9}$ all of receiver R's queries during the commitment phase will be answered zero. But putting the oracle queries aside, the hash value h does not carry any information about the bit-message b and therefore the scheme is $(1 - 2^{n/8})$-hiding.

Equivocability. Let τ be the view of an arbitrary receiver $\widehat{R}$ after the commitment phase. Let (A, h) be the matrix sent by $\widehat{R}$ and the hash value received from S, and let $(x_1, y_1), \ldots, (x_t, y_t)$ be the oracle query/answer pairs asked from f_b and described in τ (where b is the input given to the sampler algorithm Sam). For each x_i let $Z_i = \{z \mid y_i \in I(z) \wedge zA = h\}$ be the set of possible values for a which y_i falls into $I(z)$, and recall that $Z = \{z \mid zA = h\}$. (As we said before, since the last m rows of A are independent, $|Z_i| \leq 2$ for all i.) Note that $y_i = 0$ if and only if $a \notin Z_i$. Therefore the set of possible values for a is $W = Z \cap_{y_i \neq 0} Z_i \setminus \cup_{y_i = 0} Z_i$. Thus the sampler Sam simply outputs a random element of W as the value for a. This can be done efficiently since any candidate for a (and in particular the members of W) can be identified by their first $n - m$ bits, and the simulator is given the list $\cup_{y_i = 0} Z_i$ of impossible values for a which is of length $O(t)$, and therefore the verifier can finish the sampling in time $\text{poly}(|\tau|) \geq \text{poly}(t)$ which is allowed. $\square$

6.4.3 Efficiency of Noninteractive Locking Schemes

By a *noninteractive locking scheme* (NLS), we mean an ILS where the commitment phase is noninteractive and sender S only participates in the decommitment phase. Note that an efficient locking scheme by definition uses poly(n)-sized circuits to implement the locking oracle σ, and therefore σ can have at most poly(n) entropy. In this section we show that there exist no efficient NLS with super-polynomial security.

Since we are going to prove that NLS's cannot be efficient, we need to deal with unbounded senders. Thus we can no longer assume that the decommitment phase is only a message (b, r_S) sent to the receiver, because the randomness r_S used by the sender can be exponentially long. Therefore to prove the strongest possible *negative* result, we allow the decommitment phase of a NLS to be interactive.

The following theorem clearly implies Part 2 of Theorem 6.4.1.

Theorem 6.4.13. *Let $\Lambda = (S, \sigma, R)$ be any NLS for message space $\{0,1\}$ in which the function σ of the locking oracle has Shannon entropy at most $H(\sigma) \leq \frac{uq}{1000}$ when the committed bit b is chosen at random $b \leftarrow \{0,1\}$. Let u be an upper-bound on the number of oracle queries to σ asked by the receiver R in the decommitment phase. Then either of the following holds:*

- **Violation of binding:** *There is a fixed locking oracle $\widehat{\sigma}$, and a sender strategy $\widehat{S}$ such that when $\widehat{\sigma}$ is used as the locking orale, for both $b = 0$ and $b = 1$, $\widehat{S}$ can decommit successfully with probability at least $4/5$.*

- **Violation of hiding:** *There exists an unbounded receiver $\widehat{R}$ who can guess the random bit $b \leftarrow \{0,1\}$ used by (S, σ) with probability at least $4/5$ by asking at most u queries to the locking oracle σ.*

Intuition behind Theorem 6.4.13. Our main tool in proving Theorem 6.4.13 is the notion of "canonical entropy learner" (EL) introduced in Definition 6.4.14. Roughly speaking, EL is an efficient-query (computationally unbounded) algorithm which learns a randomized function f (with an oracle access to f) under the uniform distribution assuming that f has a bounded amount of entropy. EL proceeds by choosing to ask one of the "unbiased" queries of f at any step and stop if such queries do not exist. An unbiased query x is one whose answer $f(x)$ is not highly predictable with the current knowledge gathered about f by EL. Whenever EL chooses to ask a query it learns non-negligible entropy of f, and thus the process will stop after poly(n) steps. On the other hand, when EL stops, all the remaining queries are biased and thus will have a predictable answer *over the randomness of* f. We prove that either the receiver is able to find out the secret message of the sender (in an NLS) by running the EL algorithm, or otherwise if by the end of the learning phase still part of the entropy left in the locking oracle is hiding the secret message, then a malicious prover can plant at least two different messages in the locking oracle in such a way that it can decommit to successfully.

Notation. In the following, for clarity, we use bold letters to denote random variables and use the non-bold version of the bold variables to denote the samples of that random variable: $x \leftarrow \mathbf{x}$.

Definition 6.4.14 (Canonical entropy learner)**.** Suppose $\mathbf{f}$ is a random variable with the support set $\{f \mid f\colon \{0,1\}^n \to \{0,1\}\}$ of all Boolean functions defined over n bits of input. The *canonical entropy learner* $\mathsf{EL}_\epsilon^{\mathbf{f}}$ is an oracle algorithm with access to $\mathbf{f}$ which asks its queries as follows. At any time there is a set $\mathbf{Q}$ of query/answer pairs that $\mathsf{EL}_\epsilon^{\mathbf{f}}$ knows about $\mathbf{f}$. Then if there is any new query x (outside of $\mathbf{Q}$) for which $\epsilon \leq \Pr[\mathbf{f}(x) = 0 \mid \mathbf{Q}] \leq 1 - \epsilon$ (in which case we call x an ϵ-unbiased query of $\mathbf{f}$ with

respect to $\mathbf{Q}$) then $\mathsf{EL}_\epsilon^{\mathbf{f}}$ asks the lexicographically first such x. $\mathsf{EL}_\epsilon^{\mathbf{f}}$ continues asking as long as there is any ϵ-unbiased queries left. We call a query x $(1-\epsilon)$-biased (with respect to $\mathbf{Q}$), if it is not ϵ-unbiased.

Lemma 6.4.15. *Suppose $\epsilon < 1/2$ and let the random variable $\mathbf{Q}$ be the set of query/answer pairs that $\mathsf{EL}_\epsilon^{\mathbf{f}}$ learns when interacting with $\mathbf{f}$. Then it holds that $\mathrm{E}[|\mathbf{Q}|] \leq H(\mathbf{f})/\epsilon$ where $H(\cdot)$ is the Shannon entropy.*

Proof. When x is an ϵ-unbiased query of $\mathbf{f}$ with respect to $\mathbf{Q}$, then conditioned on $\mathbf{Q}$, $\mathbf{f}(x)$ has Shannon entropy at least $H(\mathbf{f}(x) \mid \mathbf{Q}) \geq \epsilon \log(1/\epsilon) + (1-\epsilon)\log(1/(1-\epsilon)) > \epsilon \log(1/\epsilon) > \epsilon$. Suppose $x_1, \ldots, x_{|\mathbf{Q}|}$ are the queries asked by $\mathsf{EL}_\epsilon^{\mathbf{f}}$ and $x_{|\mathbf{Q}|+1}, \ldots, x_{2^n}$ are the remaining queries not asked by $\mathsf{EL}_\epsilon^{\mathbf{f}}$ in the lexicographic order, and suppose $\mathbf{Q}_i$ is the set containing the pairs of the queries $x_1, \ldots, x_i$ joint with their answers. Since the information about $x_1, \ldots, x_i$ and $\mathbf{f}(x_1), \ldots, \mathbf{f}(x_i)$ is encoded in $\mathbf{Q}_{i-1}$, therefore it holds that $H(\mathbf{f}) = \sum_{i \in [2^n]} H(\mathbf{f}(x_i) \mid \mathbf{Q}_{i-1})$. Note that

$$H(\mathbf{f}(x_i) \mid \mathbf{Q}_{i-1}) = \Pr[|\mathbf{Q}_i| \geq i] H(\mathbf{f}(x_i) \mid \mathbf{Q}_{i-1} \wedge |\mathbf{Q}_i| \geq i) + \Pr[|\mathbf{Q}_i| < i] H(\mathbf{f}(x_i) \mid \mathbf{Q}_{i-1} \wedge |\mathbf{Q}_i| < i) \ .$$

We claim that $H(\mathbf{f}(x_i) \mid \mathbf{Q}_{i-1} \wedge |\mathbf{Q}_i| \geq i) \ \geq \epsilon$. The reason is that we are conditioning on the cases of $\mathbf{Q}_{i-1}$ which determine x_i as an ϵ-unbiased point (and we do not condition on any more information). Therefore it holds that

$$\begin{aligned} H(\mathbf{f}) &= \sum_{i \in [2^n]} H(\mathbf{f}(x_i) \mid \mathbf{Q}_{i-1}) \\ &\geq \sum_{i \in [2^n]} \Pr[|\mathbf{Q}_i| \geq i] H(\mathbf{f}(x_i) \mid \mathbf{Q}_{i-1} \wedge |\mathbf{Q}_i| \geq i) \\ &\geq \sum_{i \in [2^n]} \Pr[|\mathbf{Q}_i| \geq i] \epsilon = \epsilon \, \mathrm{E}[|\mathbf{Q}|], \end{aligned}$$

and so $\mathrm{E}[|\mathbf{Q}|] \leq H(\mathbf{f})/\epsilon$. □

Now we turn to proving Theorem 6.4.13.

Proof. (of Theorem 6.4.13)

For simplicity, and without loss of generality, we assume that the oracle σ is Boolean and is only defined over inputs of length n.

The attack of $\widehat{R}$ is as follows. The receiver runs the canonical entropy learner $\mathsf{EL}^{\mathbf{f}}_{\epsilon}$ with $\epsilon = 1/100u$ over the (random) function of the oracle σ (where the randomness of σ comes from the randomness r_S shared with the sender which includes the random bit $b \leftarrow \{0,1\}$). Recall that the receiver cannot ask more than u queries. We define $\mathbf{Q}_u$ to be the set of query/answer pairs that the receiver (who asks at most u oracle queries) learns about σ. It always holds that $\mathbf{Q}_u \subseteq \mathbf{Q}$ where $\mathbf{Q}$ is the set of query/answer pairs that $\mathsf{EL}^{\mathbf{f}}_{\epsilon}$ would learn without stopping after u queries. We call a set T of query/answer pairs for σ *predicting* if there exists $c \in \{0,1\}$ such that $\Pr[b = c \mid T] \geq 9/10$. Let Q_u be the value sampled by the random variable $\mathbf{Q}_u$ and Q be the value sampled by the random variable $\mathbf{Q}$. There are two possibilities.

(1) There is a possible Q_u (as a value of $\mathbf{Q}_u$) where Q_u is not predicting and $Q = Q_u$ (i.e. there is no ϵ-unbiased query conditioned on Q_u).

(2) All possible values of Q_u (as the result of the receiver's learning algorithm) are either predicting or $Q \neq Q_u$.

In the following we prove that Case **(1)** implies the violation of binding and Case **(2)** implies the violation of hiding according to Theorem 6.4.13.

Case (1). First suppose that Case **(1)** holds. In particular there is a possible value Q as the canonical entropy learner's knowledge about σ such that Q is not predicting,

namely:

$$1/10 < \Pr[b = 0 \mid Q] < 9/10 \quad \text{and} \quad 1/10 < \Pr[b = 1 \mid Q] < 9/10 \ .$$

Also note that conditioned on Q all the points x out of Q are $(1-\epsilon)$-biased and thus have a unique "likely answer". Namely for each x there is l_x such that $\Pr[\sigma(x) \neq l_x \mid Q] < 1/100u$. Since Q is not predicting, for all x it holds that:

$$\Pr[\sigma(x) \neq l_x \mid Q \wedge b = 0] = \frac{\Pr[\sigma(x) \neq l_x \wedge b = 0 \mid Q]}{\Pr[b = 0 \mid Q]} < \frac{\Pr[\sigma(x) \neq l_x \mid Q]}{1/10} < 1/10u \ .$$

(The same also holds if we condition on $b = 1$.) Therefore in addition to Q, if we condition on $b = 0$ or $b = 1$, still all the points in σ are $(1 - 1/10u)$-biased.

Now we show that the binding property could be violated by a malicious sender and oracle $(\widehat{S}, \widehat{\sigma})$. Consider the following cheating strategy:

1. Distribution of $\widehat{\sigma}$: Sample two oracles along with their (possibly exponentially long) randomness $(\sigma_0, r_0), (\sigma_1, r_1)$ according to the distributions $(\sigma_0, r_0) \leftarrow (\sigma, r_S \mid Q \wedge b = 0)$ and $(\sigma_1, r_1) \leftarrow (\sigma, r_S \mid Q \wedge b = 1)$ and define the oracle $\widehat{\sigma}$ as follows.

   - If $\sigma_0(x) = \sigma_1(x) = l_x$, then $\widehat{\sigma}(x) = \sigma_0(x) = \sigma_1(x)$.
   - If $\sigma_0(x) \neq l_x$ but $\sigma_1(x) = l_x$, then $\widehat{\sigma}(x) = \sigma_0(x)$.
   - If $\sigma_1(x) \neq l_x$ but $\sigma_0(x) = l_x$, then $\widehat{\sigma}(x) = \sigma_1(x)$.
   - If $\sigma_0(x) = \sigma_1(x) \neq l_x$, then $\widehat{\sigma}(x) = \sigma_0(x) = \sigma_1(x)$.

2. The algorithm of $\widehat{S}$: To decommit to $b = 0$, the sender $\widehat{S}$ announces $b = 0$ and uses r_0 as the randomness of S to interact in the decommitment phase. To decommit to $b = 1$, $\widehat{S}$ acts similarly by announcing $b = 1$ and using the

randomness r_1 in its interaction of the decommitment phase.

Claim 6.4.16. *If the sender generates the locking oracle $\widehat{\sigma}$ (from σ_0, σ_1) as above and decommits to zero by announcing 0 and using the randomness r_0 in the decommitment phase, then the receiver will accept the interaction with probability at least 9/10.*

Proof. After running the decommitment to zero, we call the experiment a defeat if the receiver asks any query x such that $\sigma_1(x) \neq l_x$, and will call it a win otherwise. Note that if the experiment is a win, then the receiver will accept the decommitment because all of the answers he gets for his queries from $\widehat{\sigma}$ will agree with σ_0.

Now consider another experiment in which $\widehat{\sigma}$ is again generated exactly the same as above, but now σ_0 is used during the decommitment phase. It is clear that now the receiver will accept, but again, we will call the experiment a defeat if the receiver asks any query x (from σ_0) such that $\sigma_1(x) \neq l_x$.

We claim that the probability that the first experiment is a defeat is exactly the same as that of the second experiment. The reason is that, although the receiver in general behaves differently in the two experiments, the difference starts by asking a query x such that $\sigma_1(x) \neq l_x$, and otherwise all the answers given to to the receiver will be the same between $\widehat{\sigma}$ and σ_0.

Now we claim that the probability of defeat in the second experiment is less than 1/10. It would finish the proof since in that case the probability of defeat in the first experiment also will be less than 1/10, and therefore the receiver will accept in the first experiment with probability more than 9/10.

In the second experiment we can pretend that σ_1 is sampled *after* the verification is done using the oracle σ_0. The reason is that the behavior of the receiver is defined only based on σ_0, while σ_1 is only used to determine whether or not the experiment ends in defeat. So suppose $X = \{x_1, \ldots, x_u\}$ be the queries of the receiver before

sampling σ_1. Then for each one of $x_i \in X$ it holds that $\Pr[\sigma_1(x) \neq l_x] < 1/10u$ and by the union bound, the probability that for at least one of the $x \in X$, we get $\sigma_1(x) \neq l_x$ is less than $1/10$. $\square$

By symmetry, the same argument shows that if $\widehat{\sigma}$ is generated as above and the sender pretends that σ_1 is in the oracle and decommits to $b = 1$ (by using r_1) the process leads to the accept of the receiver R with probability more than $9/10$.

We call a *fixed* $(\widehat{\sigma}, r_0, r_1)$ generated as above a good sample for $b = 0$ if the decommitment of $\widehat{S}$ to $b = 0$ as above leads to accept with probability at least $4/5$. A simple average argument shows that the sampled $(\widehat{\sigma}, r_0, r_1)$ is good for $b = 0$ with probability more than $1/2$. A similar argument shows that the sampled $(\widehat{\sigma}, r_0, r_1)$ is also good for $b = 1$ with probability more than $1/2$. By the pigeonhole principle there is a fixed $(\widehat{\sigma}, r_0, r_1)$ which is both good for $b = 0$ and $b = 1$. Using this $(\widehat{\sigma}, r_0, r_1)$ the sender can decommit to both $b = 0$ and $b = 1$ and succeed with probability at least $4/5$.

Case (2). If Case **(2)** holds, then we claim that the receiver R can guess the bit b with probability at least $4/5$. The reason is that by Lemma 6.4.15 it holds that $\mathrm{E}[|\mathbf{Q}|] \leq H(\sigma)/\epsilon \leq u/10$. Therefore by Markov inequality, with probability at least $9/10$ it holds that $|\mathbf{Q}| \leq u$ in which case $Q_u = Q$. Moreover since Case **(2)** holds, whenever $Q = Q_u$, then Q is predicting and the receiver R can guess b correctly with probability at least $9/10$. Therefore the receiver R would guess b correctly with probability at least $(9/10)(9/10) > 4/5$ which is a violation of hiding according to Theorem 6.4.13. $\square$

6.5 The Impossibility of Oblivious Transfer

In this section we prove that in the stateless token model, there is no statistically secure protocol for Oblivious Transfer (OT), when the only limitation on malicious parties is being bounded to make polynomially many queries to the tokens. We first define the (bit) oblivious transfer functionality. For simplicity we use a definition with completeness one, but our negative result easily extends to protocols with imperfect completeness.

The stateless token model. In the stateless (tamper-proof hardware) token model, two (computationally unbounded) interactive algorithms A and B will interact with the following extra feature to the standard model. Each party at any time during the protocol can construct a circuit T and put it inside a "token" and send the token T to the other party. The party receiving the token T will have *oracle access* to T and is limited to ask poly(n) number of queries to the token. The parties can exchange poly(n) number of tokens during the interaction. The stateless token model clearly extends the IPCP model in which there is only one token sent from the prover to the verifier in the beginning of the game. Therefore proving any *impossibility* result in the stateless token model clearly implies the same result for the the IPCP model. It is easy to see that without loss of generality the parties can avoid sending "explicit messages" to each other and can only use tokens (with messages planted inside the tokens) to simulate all the classical communication with the tokens.

Definition 6.5.1 (Oblivious Transfer)**.** A protocol (S, R) for oblivious transfer (in the standard or the token model), consists of a sender S and a receiver R. The parties both receive 1^n where n is the security parameter. The sender S holds the secret input $(x_0, x_1), x_i \in \{0, 1\}$, and the receiver R holds the input $i \in \{0, 1\}$. At the end of the protocol, the receiver outputs y where (with probability one) it holds that $y = x_i$.

We define the following security measures.

- **Receiver's security:** This property guarantees that the sender is not able to find out which of x_0 and x_1 is read by the receiver. Formally, for any malicious sender $\widehat{S}$ who asks $\mathrm{poly}(n)$ token queries and outputs j, it holds that $\Pr_{i \leftarrow \{0,1\}}[i = j] \leq 1/2 + \mathrm{neg}(n)$.

- **Sender's security:** This property guarantees that no receiver can read both of x_0 and x_1. Formally, for any malicious receiver $\widehat{R}$ who asks $\mathrm{poly}(n)$ token queries, with probability $1 - \mathrm{neg}(n)$ the view $v_{\widehat{R}}$ of $\widehat{R}$ satisfies either of the following:

$$\Pr_{(x_0,x_1)\leftarrow\{0,1\}^2}[x_0 = 0 \mid v_{\widehat{R}}] = \frac{1}{2} \pm \mathrm{neg}(n) \quad \text{or} \quad \Pr_{(x_0,x_1)\leftarrow\{0,1\}^2}[x_1 = 0 \mid v_{\widehat{R}}] = \frac{1}{2} \pm \mathrm{neg}(n).$$

Oblivious transfer by semi-honest parties. If one of the parties is semi-honest (i.e. runs the protocol honestly, and only remember's its view for further off-line investigation), then in fact unconditionally secure OT *is* possible in the stateless token model. If the receiver is honest, then the protocol is simply a token T sent from the sender which encodes $T(0) = x_0, T(1) = x_1$. The receiver will read $T(i)$ to learn x_i. Moreover it is well known that secure OT in one direction implies the existence of secure OT in the other direction, so if the sender is semi-honest unconditionally secure OT is possible in the stateless token model.

In this section we prove that statistically secure OT is impossible in the stateless token model, if both parties are *slightly* more malicious than just being semi-honest. Roughly speaking, we define the notion of "curious" parties who run the original protocol (honestly), but will ask more queries from the tokens along the way.[6] We

[6]The term "honest but curious" is sometimes used equivalent to "semi-honest". Our notion is different from both of them because a curious party deviates from the protocol slightly by learning more but emulates the original protocol honestly.

will prove that for any protocol (A, B) aiming to implement OT, there are curious extensions of the original parties $(A_{\mathsf{cur}}, B_{\mathsf{cur}})$ who break the security of the protocol. More formally we define curious parties as follows.

Definition 6.5.2. Let (A, B) be a two party protocol in the token model. We call the protocol A_{cur} a *curious* extension of A, if A_{cur} runs the same protocol as A, but at any point during the protocol it might ask arbitrary queries from tokens sent from B. We call A_{cur} efficient, if it asks only poly(n) queries totally. By $Q_{A_{\mathsf{cur}}}$ we denote the set of triples representing total query/answer information that A_{cur} gathers about the tokens it receives, and we use Q_A to denote only the query/answers for the underlying emulated algorithm A (and so $Q_A \subseteq Q_{A_{\mathsf{cur}}}$).

We will prove the following theorem.

Theorem 6.5.3 (No unconditional OT from stateless tokens)**.** *Let (S, R) be any protocol for the oblivious transfer in the stateless token model. Then there are curious extensions $(S_{\mathsf{cur}}, R_{\mathsf{cur}})$ to the original algorithms where $(S_{\mathsf{cur}}, R_{\mathsf{cur}})$ (and thus (S, R)) is not a secure protocol for oblivious transfer even when the inputs are random. More formally either of the following holds:*

- **Violation of sender's security:** *When the sender S chooses x_0 and x_1 at random from $\{0, 1\}$ and interacts with R_{cur}, then R_{cur} can find out both of x_0 and x_1 with probability at least* $51/100$.
- **Violation of receiver's security:** *When the receiver R chooses $i \leftarrow \{0, 1\}$ at random and interacts with S_{cur}, then S_{cur} can guess i correctly with probability at least* $51/100$.

In the following section we first show that any protocol (S, R) with the following sampling condition can not implement OT securely (Lemma 6.5.5). The sampling

condition is that one can sample the randomness of the receiver R conditioned on the view of the sender S. This sampling condition extends the "accessible entropy" notion introduced by Haitner et al [86] to the token model (in an information theoretic way) and is equivalent to saying that the entropy of the receiver is accessible. Then we show (Theorem 6.5.8) that for any protocol (S, R), there is a curious extension $S_{\mathsf{cur}}, R_{\mathsf{cur}}$ where almost all the entropy is accessible, and this will prove Theorem 6.5.3.

Notation. In the following, for clarity, we use bold letters to denote random variables and use the non-bold version of the bold variables to denote the samples of that random variable: $x \leftarrow \mathbf{x}$. For example $\mathbf{r}_A$ will denote the randomness of Alice as a random variable and r_A denotes the actual randomness used by Alice.

Breaking Oblivious Transfer by Accessing the Entropy

In the standard model of interaction (without tokens) all the information exchanged between two parties is represented in the transcript τ of the interaction. This means that information theoretically, an unbounded Bob can sample from $r \leftarrow (\mathbf{r}_B \mid \tau)$: his own space of randomness conditioned on what Alice knows about it (which is the transcript of the protocol τ) and vice versa. So information theoretically Bob is not committed to anything more than what Alice knows about him (but of course Bob might not be able to sample from this distribution *efficiently*). Roughly speaking if the sampling $r_B \leftarrow (\mathrm{Supp}(\mathbf{r}_B) \mid \tau)$ can be done efficiently at any time during the protocol, [86] calls such protocol one with "accessible entropy".

The fact that in the standard model unbounded parties can access all the entropy makes unconditional oblivious transfer to be impossible in this model. The reason is that at the end of the protocol, either the sender S can find out which bit is read by the receiver R by sampling from $r \leftarrow (\mathbf{r}_R \mid \tau)$, or otherwise a sample from $r \leftarrow (\mathbf{r}_R \mid \tau)$

will reveal either of the inputs x_0, x_1 with probability $\approx 1/2$ and thus Bob can find out both of them by taking enough number of samples. We will formalize this argument in the case of token model in Lemma 6.5.5 below.

In the token model, however, the information exchanged between the parties is not "shared" by them. Namely, the set Q_A (see Definition 6.5.2) captures the information that the algorithm A gathers from the tokens he has received from the other party B, and this set is different from Q_B. Moreover, it might be impossible for A to find out Q_B exactly. Therefore if we consider $\tau = (Q_A, Q_B)$ to be the transcript of the protocol in the token model, Alice and Bob each know part of τ, and this makes the above argument (for the impossibility of OT) fail. In fact, as we said, if either of the parties is semi-honest, then unconditional OT *is* possible in the stateless token model.

The following definition generalizes the notion of accessible entropy to the token model (from an information theoretic perspective).

Definition 6.5.4 (Accessible entropy in the token model)**.** Let (A, B) be a protocol in the token model between Alice and Bob. We say the entropy of Bob can be $(1-\delta)$-accessed if there is a sampling algorithm Sam_B as follows. Sam_B receives a possible view for Bob (r, Q) as input and outputs another possible view for Bob (r', Q') with the following property: With probability at least $1-\delta$ over the choice of $(r_A, r_B) \leftarrow (\mathbf{r}_A, \mathbf{r}_B)$, if Q_A, Q_B denote to the information that Alice and Bob has gathered from the tokens *at any time during* the execution of the protocol, it holds that

$$\mathsf{SD}((\mathbf{r}_B, \mathbf{Q}_B \mid r_A, Q_A), \mathsf{Sam}_B(r_B, Q_B)) \leq \delta.$$

We say that B has δ inaccessible entropy, if the entropy of B is not $(1-\delta)$-accessible.

Comments about Definition 6.5.4 .

One can define a computational version of Definition 6.5.4 in the standard model where Sam_B is efficient and given τ it samples from a distribution which is δ-close to $(\mathbf{r}_B \mid \tau)$. It can be shown that an interactive algorithm B has non-negligible inaccessible entropy in this definition if and only if B has non-negligible inaccessible entropy according to the definition of [86].

[86] shows that in the standard computational setting non-negligible inaccessible entropy implies statistically hiding commitments. We shall point out that in the stateless token model, as we will see, if both parties are "curious enough", then almost all the entropy can be accessed, but as we saw in Theorem 6.4.1 unconditionally secure commitment schemes *does* exist. The difference between the standard computational model of interaction and the token model is that in the standard computational model the sampled $r \leftarrow (\mathbf{r}_B \mid \tau)$ can be used (by a potentially malicious Bob) as a witness that Bob is running the protocol appropriately and even be used to continue the protocol without being caught by Alice, but in the token model a sampled $r = \mathsf{Sam}_B(r_B, Q_B)$ might contradict the future queries of Alice to the tokens she that is holding from Bob. The fact that Alice can caught Bob's claim by asking more token queries is highly used in our both Constructions 6.4.4 and 6.4.9 of ILS's. In fact, if one defines a *simple commitment* scheme as one where the receiver is not allowed to ask token queries during the decommitment phase, then a similar argument to Lemma 6.5.5 below shows that simple commitment schemes imply (non-negligible) inaccessible entropy, and therefore unconditional simple commitments are impossible to achieve in the stateless token model.

Lemma 6.5.5 (Oblivious transfer needs inaccessible entropy)**.** *Let (S, R) be a protocol for* random *oblivious transfer where the sender's inputs x_0, x_1 are part of r_S and receiver's input i is part of r_B. If (S, R) has perfect completeness and the entropy of*

the receiver R can be 0.99*-accessed, then either of the following holds:*

- **Violation of sender's security:** *When the sender S interacts with R, then R can find out both of x_0 and x_1 correctly with probability at least* 51/100*.*
- **Violation of receiver's security:** *When the receiver R chooses $i \leftarrow \{0,1\}$ at random and interacts with S, then S can guess i correctly with probability at least* 51/100*.*

Proof. We show that if Sam_R 1-accesses the entropy, then either of the security failures above holds with probability at least 53/10. Using the actual Sam_R (which 0.99-accesses the entropy) will change the success probability of the attacks by at most 2/100 which still remain at least 51/100. So in the following we assume that Bob also can sample from the distribution $(\mathbf{r}_R \mid r_S, Q_S)$ where by Q_S we denote the final set of query/answers that the sender learns about the receiver's tokens. We call (r_S, Q_S) a predicting pair, if either $\Pr[i = 0 \mid r_S, Q_S] \geq 9/10$, or $\Pr[i = 1 \mid r_S, Q_S] \geq 9/10$. Now there are two possible cases:

- $\Pr[(r_S, Q_S)$ is predicting$] \geq 1/10$: Then the sender can predict i by probability at least $1/2 + 1/10 > 53/10$ as follows: Whenever (r_S, Q_S) is predicting, use the predicted value, and whenever it is not use a random guess. This violates the security of the receiver.
- $\Pr[(r_S, Q_S)$ is predicting$] \leq 1/10$: Therefore, with probability at least 9/10 (r_S, Q_S) is not predicting. For a non-predicting (r_S, Q_S), by sampling $(r'_R, Q'_R) \leftarrow (\mathbf{r}_R, \mathbf{Q}_R \mid r_S, Q_S)$ we get either of the cases $i = 0$ or $i = 1$ with probability at last 1/10. By the assumed perfect completeness of the protocol the output of the sampled (r'_R, Q'_B) is equal to x_i. Note that sender can sample from $(\mathbf{r}_R, \mathbf{Q}_R \mid r_S, Q_S)$ by using the sampler Sam_R. When (r_S, Q_S) is not predicting,

by sampling n samples $(r'_R, Q'_R) \leftarrow (\mathbf{r}_R, \mathbf{Q}_R \mid r_S, Q_S)$, the receiver can find both of x_0 and x_1 with probability $1 - \text{neg}(n)$. Therefore, since (r_S, Q_S) is not predicting with probability $9/10$, the receiver can find both inputs of the sender with probability at least

$(9/10) \cdot (9/10 - \text{neg}(n)) > 53/100$.

□

Curious Extensions Who Access the Entropy

In this section we define a specific curious extension to the protocols in the stateless token model which we call "canonical" curious extensions and prove that almost all the entropy of such protocols can be accessed. Roughly speaking, a canonical curious extension of an interactive algorithm is a curious extension of the original protocol where the parties ask their "extra" token queries according to the canonical entropy learner of Definition 6.4.14. Each time that an unbiased query q is asked, it reveals a noticeable amount of information about the other party's private randomness. By the same argument as Lemma 6.4.15, since the entropy $H(\mathbf{r}_A) + H(\mathbf{r}_B) \leq \text{poly}(n)$ is bounded, by asking only efficient number of queries one can learn enough information to predict the answer to any other (fixed) query to the tokens. In particular, the answer to the query that is going to be asked in the emulation of the original protocol will be predictable, so we can ignore asking such queries and use the predicted answers in the emulation of the original protocol. We call the new protocol which uses the predicted values an "ideal" protocol $(A_{\mathsf{id}}, B_{\mathsf{id}})$ which will simulate the original protocol (A, B) up to $1/\text{poly}(n)$ statistical distance.

Finally, the main point is that in the ideal protocol the parties know *exactly* what the other party knows about them! The proof uses induction. Assuming that Alice and Bob both know $(Q_{A_{\mathsf{id}}}, Q_{B_{\mathsf{id}}})$, they can find out which queries are ϵ-unbiased from

Alice's and Bob's point of view. When Alice is asking such a query Bob knows the query *and* its answer (since it is from a token generated by Bob) and vice versa. Therefore, $(r_{A_{\mathsf{id}}}, Q_{A_{\mathsf{id}}})$ will determine $Q_{B_{\mathsf{id}}}$ and $(r_{B_{\mathsf{id}}}, Q_{B_{\mathsf{id}}})$ will determine $Q_{A_{\mathsf{id}}}$ and in a sense we are back to the standard model where the transcript is known to both parties, and thus all the entropy can be accessed!

It will be easier for us to first define the ideal experiment Ideal and then define how the canonical curious parties behave.

Definition 6.5.6 (Ideal experiment.)**.** Let (A, B) be a protocol in the stateless token model. The protocol $(A_{\mathsf{id}}, B_{\mathsf{id}})$ will depend on (A, B) and will be executed in the experiment Ideal as follows.

- **Randomness:** $(A_{\mathsf{id}}, B_{\mathsf{id}})$ will use the same randomness (r_A, r_B) and will (try to) emulate (A, B).
- **Set of token query/answers:** $(A_{\mathsf{id}}, B_{\mathsf{id}})$ will inductively update the sets $Q_{A_{\mathsf{id}}}$ and $Q_{B_{\mathsf{id}}}$ which encode the information they have gathered from the other party's tokens so far.
- **Mutual Knowledge:** A crucial point is that $Q_{A_{\mathsf{id}}}$ and $Q_{B_{\mathsf{id}}}$ will be known to *both* Alice and Bob inductively. Namely, A_{id} can find out the updates to the set $Q_{B_{\mathsf{id}}}$ without B_{id} explicitly revealing that information to A_{id} and vice versa. How this condition holds and how the emulation of (A, B) is done is clarified next.
- **Emulation:** Suppose during the emulation we are at a time where A needs to ask the query q_0 from a token T. Instead of doing so, A_{id} runs the canonical entropy learner of Definition 6.4.14 over the tokens $(T_1^B, \ldots, T_i^B)$ that he holds from Bob. Namely, A_{id} considers the uniform distribution of $(\mathbf{r}_A, \mathbf{r}_B)$ consistent

with the "transcript" of the interaction: $(Q_{A_{\mathsf{id}}}, Q_{B_{\mathsf{id}}})$. Note that, by induction's assumption A_{id} knows $Q_{B_{\mathsf{id}}}$ as well. Now if there is any query q to any of the tokens $(T_1^B, \ldots, T_i^B)$ sent from Alice such that q is ϵ-unbiased, then A_{id} asks the lexicographically first such query q_1 and updates $Q_{A_{\mathsf{id}}}$. We stress that since B_{id} knows also $Q_{A_{\mathsf{id}}}$, he is also able to find the query q_1, and moreover since this query is asked to one of the tokens that Bob has generated, B_{id} knows the answer as well. So Bob also knows the update to the set $Q_{A_{\mathsf{id}}}$ the same as Alice. When A_{id} is done with the learning phase before asking q_0, it *not* ask the query q_0, but rather will use the likely answer l_{q_0} which is uniquely determined by the updated $(Q_{A_{\mathsf{id}}}, Q_{B_{\mathsf{id}}})$ (but of course l_{q_0} might be different from the actual solution provided by the token). B_{id} will emulate B similarly.

Now we define the experiment CanCur in which a canonical curious extension of the protocol (A, B) is defined.

Definition 6.5.7 (Canonical curious extensions)**.** Let (A, B) be a two party protocol in the stateless token model. A *Canonical Curious* extension of A denoted by A_{cc} receives two parameters ϵ: the curiosity parameter and u: the upper-bound on the number of extra queries asked by A. A_{cc} acts in the experiment CanCur as follows:

- A_{cur} runs the algorithm A with randomness r_A and emulates the interaction of A with B. The set Q_A denotes the set of token query/answers corresponding to this emulation.

- A_{cur} also keeps updating a set of token query/answers $Q_{A_{\mathsf{id}}}$ by *pretending* that it is being executed in the experiment Ideal. A_{cur} stops asking more extra queries (i.e. the queries out of Q_A) whenever $|Q_{A_{\mathsf{id}}}| = u$. Note that in CanCur (as opposed to CanCur), A_{cc} *does* ask the emulated queries of A from the tokens and uses the answer for further computations.

- The total token queries/answer knowledge of A_{cur} is encoded in $Q_{A_{\mathsf{cur}}} = Q_A \cup Q_{A_{\mathsf{id}}}$.

Now we prove the following theorem showing that by taking ϵ small enough, one can access almost all the entropy of $(A_{\mathsf{id}}, B_{\mathsf{id}})$ efficiently.

Theorem 6.5.8 (Canonical curious parties can access the entropy)**.** *Let (A, B) be a protocol in the stateless token model between Alice and Bob where they totally ask $k = \text{poly}(n)$ number of queries to the tokens exchanged. For a given $\delta = 1/\text{poly}(n)$ let ϵ be such that $\delta = 2(\epsilon k + \sqrt{\epsilon k})$ and let $u = \frac{|r_A|}{k\epsilon^2}$. If ϵ is used as the curiosity parameter and u is used as the upper-bound parameter, then the entropy of B_{cc} in $(A_{\mathsf{cc}}, B_{\mathsf{cc}})$ is $(1-\delta)$-accessible.*

Note that Theorem 6.5.8 and Lemma 6.5.5 together imply Theorem 6.5.3 as a corollary.

Intuition. Roughly speaking Theorem 6.5.8 follows from the following facts:

1. **Entropy is accessible in Ideal:** The entropy of A_{id} is 1-accessible, because $Q_{A_{\mathsf{id}}}$ and $Q_{B_{\mathsf{id}}}$ are known to both parties at any time during the protocol.

2. **Parties are efficient in Ideal:** By Lemma 6.4.15 A_{id} on average asks at most $H(\mathbf{r}_B)/\epsilon = |\mathbf{r}_B|/\epsilon \leq \text{poly}(n)$ number queries.

3. **Ideal and CanCur are close:** As long as the likely answers used by A_{id} and B_{id} for the emulation of (A, B) are correct, the two experiments Ideal and CanCur are *exactly* the same. Let WG (Wrong Guess) be the event that one of these likely answers used is incorrect. By definition, he probability that WG happens at any particular query in Ideal is at most ϵ, and therefore the probability of WG is bounded by $\epsilon \cdot k$. By taking ϵ small enough we can make $\epsilon \cdot k$ arbitrary small.

The formal proof follows.

Proof. (of Theorem 6.5.8) The sampling algorithm Sam_B in CanCur will perform the sampling by simply pretending that it is being used in the experiment Ideal (by ignoring Q_B and only using $Q_{B_{\mathsf{id}}}$ and r_B). We will show that with probability $1-\delta$ over $(\mathbf{r}_A, \mathbf{r}_B)$ the sampler Sam_B will sample δ-close to the right distribution all along the execution of CanCur.

As a mental experiment, we sample $(r_A, r_B) \leftarrow (\mathbf{r}_A, \mathbf{r}_B)$ and run *both* of the experiments CanCur and Ideal in parallel with the same randomness (r_A, r_B). We use the notation $\Pr_{\mathsf{cc}}[\cdot]$ to denote the probability of an event in CanCur and use $\Pr_{\mathsf{id}}[\cdot]$ to denote the probability of an event in Ideal.

Let WG_i to be the event in the experiments CanCur and Ideal that holds iff the i'th query q_i asked by Alice or Bob in is *not* the likely answer, and define the event $\mathsf{WG} = \bigvee_i \mathsf{WG}_i$.

We modify CanCur slightly and let the size of $Q_{A_{\mathsf{id}}}$ and $Q_{B_{\mathsf{id}}}$ to grow beyond u. Our proof will take care of this issue indirectly. By this change, it is easy to see that as long as WG does not happen the experiments CanCur and Ideal are exactly the same. In particular it holds that $\Pr_{\mathsf{cc}}[\mathsf{WG}] = \Pr_{\mathsf{id}}[\mathsf{WG}]$. Since (r_A, r_B) determines the whole execution of the experiments CanCur and Ideal we can talk about the event $\mathsf{WG}(\mathbf{r}_A, \mathbf{r}_B)$, and $\Pr_{(\mathbf{r}_A, \mathbf{r}_B)}[\mathsf{WG}]$ is the same in CanCur and Ideal.

The following claim shows that by taking ϵ small enough CanCur and Ideal will become statistically close.

Claim 6.5.9. *It holds that* $\Pr_{(\mathbf{r}_A, \mathbf{r}_B)}[\mathsf{WG}] \leq k\epsilon$.

Proof. If show that $\Pr_{\mathsf{id}}[\mathsf{WG}_i] \leq \epsilon$ the claim follows by union bound over $i \in [k]$. But $\Pr_{\mathsf{id}}[\mathsf{WG}_i] \leq \epsilon$ holds by the definition of the experiment Ideal and ϵ-unbiased queries. □

We call (r_A, r_B) a *good sample* if the following conditions hold when (r_A, r_B) is used to run the experiments:

1. WG does not happen.
2. $|Q_{B_{\mathsf{id}}}|$ and $|Q_{B_{\mathsf{id}}}|$ does not pass the upper limit u.
3. At any time during the execution if $Q_{A_{\mathsf{id}}}$, $Q_{B_{\mathsf{id}}}$, $Q_{A_{\mathsf{cc}}}$, and $Q_{B_{\mathsf{cc}}}$ denote the set of query/answers in the both experiments in that moment it holds that

$$\Pr_{\mathsf{id}}[\mathsf{WG} \text{ happened before } \mid Q_{A_{\mathsf{id}}}, Q_{B_{\mathsf{id}}}] \leq \sqrt{k\epsilon} \quad \text{and}$$
$$\Pr_{\mathsf{cc}}[\mathsf{WG} \text{ happened before } \mid Q_{A_{\mathsf{cc}}}, Q_{B_{\mathsf{cc}}}] \leq \sqrt{k\epsilon}.$$

Claim 6.5.10. $\Pr[(\mathbf{r}_A, \mathbf{r}_B) \textit{ is good }] \geq 1 - \delta$.

Proof. By Claim 6.5.9 the first property holds with probability at least $1 - k\epsilon$.

By Lemma 6.4.15 and using the Markov inequality the second property holds in Ideal with probability at least $1 - k\epsilon$. So both properties 1 and 2 hold with probability at least $1 - 2k\epsilon$.

We use Lemma 1.3.5 with the following parameters $X = (\mathbf{r}_A, \mathbf{r}_B), F = \mathsf{WG}$, $Z_i = (\mathbf{Q}_{A_{\mathsf{id}}}, \mathbf{Q}_{B_{\mathsf{id}}})$ where $(\mathbf{Q}_{A_{\mathsf{id}}}, \mathbf{Q}_{B_{\mathsf{id}}})$ their value at the i'th moment and $p = \sqrt{k\epsilon}$. Now Lemma 1.3.5 implies that the third property holds for Ideal (and similarly for CanCur) with probability at least $1 - \sqrt{k\epsilon}$.

Therefore all the properties hold for a random (r_A, r_B) with probability at least $1 - 2k\epsilon - 2\sqrt{k\epsilon} = 1 - \delta$. □

Suppose the sampled (r_A, r_B) is a good one. Then, by the fist property, all along the execution CanCur and Ideal will be the same. The second property will guarantee the efficaciously with respect to the upper-bound u.

Now suppose Sam_B is run over the input $(r_B, Q_{B_{\mathsf{cc}}})$ at an arbitrary moment during the execution of the system. By the first property, the sampler Sam_B will conclude the correct $Q_{A_{\mathsf{cc}}}$. Therefore Sam_B will sample exactly according to $(\mathbf{r}_{B_{\mathsf{id}}}, \mathbf{Q}_{A_{\mathsf{id}}} \mid r_{A_{\mathsf{id}}}, Q_{A_{\mathsf{id}}})$ in Ideal. By the third property the probability that WG has happened before is at most $\sqrt{k\epsilon}$ in either of CanCur or Ideal. Finally we note that if one samples $(r_B, Q_{B_{\mathsf{cc}}})$ conditioned on $(Q_{A_{\mathsf{id}}}, Q_{B_{\mathsf{id}}})$ *and* conditioned on $\neg\mathsf{WG}$, the samples have the same distribution in Ideal and CanCur. This means that if (r_A, r_B) is good, the sampler Sam_B will sample $2\sqrt{k\epsilon} < \delta$-close to the right distribution. This finishes the proof that Sam_B $(1-\delta)$-accesses the entropy of A_{cc} in $(A_{\mathsf{cc}}, B_{\mathsf{cc}})$. □

Bibliography

[1] *Proc.* 21*st STOC*, 1989. ACM.

[2] W. Aiello and J. Håstad. Statistical zero-knowledge languages can be recognized in two rounds. *Journal of Computer and System Sciences*, 42(3):327–345, 1991. Preliminary version in *FOCS'87*.

[3] M. Ajtai. Generating hard instances of lattice problems (extended abstract). In *STOC '96: Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 99–108, New York, NY, USA, 1996. ACM. ISBN 0-89791-785-5. doi: http://doi.acm.org/10.1145/237814.237838.

[4] M. Ajtai. The worst-case behavior of schnorr's algorithm approximating the shortest nonzero vector in a lattice. In *STOC '03*, pages 396–406, New York, NY, USA, 2003. ACM. ISBN 1-58113-674-9. doi: http://doi.acm.org/10.1145/780542.780602.

[5] A. Akavia, O. Goldreich, S. Goldwasser, and D. Moshkovitz. On basing one-way functions on np-hardness. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing (STOC)*, pages 701–710, 2006.

[6] B. Applebaum, B. Barak, and D. Xiao. On basing lower-bounds for learning on worst-case assumptions. In *Proc. FOCS '08*, pages 211–220, 2008.

[7] S. Arora and S. Safra. Probabilistic checking of proofs: A new characterization of np. *J. ACM*, 45(1):70–122, 1998.

[8] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and the hardness of approximation problems. *J. ACM*, 45(3):501–555, 1998.

[9] Babai, Fortnow, and Lund. Non-deterministic exponential time has two-prover interactive protocols. *CMPCMPL: Computational Complexity*, 1, 1991.

[10] L. Babai and S. Laplante. Stronger separations for random-self-reducibility, rounds, and advice. In *In IEEE Conference on Computational Complexity*, pages 98–104, 1999.

[11] L. Babai and S. Moran. Arthur-Merlin games: A randomized proof system and a hierarchy of complexity classes. *J. Comput. Syst. Sci.*, 36:254–276, 1988.

[12] B. Barak. How to go beyond the black-box simulation barrier. In *Proc. 42nd FOCS*, pages 106–115. IEEE, 2001.

[13] M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the First Annual Conference on Computer and Communications Security*, pages 62–73. ACM, November 1993.

[14] M. Bellare, O. Goldreich, and E. Petrank. Uniform generation of NP-witnesses using an NP-oracle. *Inf. Comput*, 163(2):510–526, 2000.

[15] M. Ben-Or, S. Goldwasser, J. Kilian, and A. Wigderson. Multi-prover interactive proofs: How to remove intractability assumptions. In *STOC*, pages 113–131, 1988.

[16] M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In *STOC*, pages 1–10, 1988.

[17] Bennett, Brassard, and Ekert. Quantum cryptography. *SCIAM: Scientific American*, 267, 1992.

[18] E. Biham, Y. J. Goren, and Y. Ishai. Basing weak public-key cryptography on strong one-way functions. In R. Canetti, editor, *TCC*, volume 4948 of *Lecture Notes in Computer Science*, pages 55–72. Springer, 2008. ISBN 978-3-540-78523-1. URL `http://dx.doi.org/10.1007/978-3-540-78524-8_4`.

[19] D. Bleichenbacher and U. M. Maurer. Directed acyclic graphs, one-way functions and digital signatures (extended abstract). In Y. G. Desmedt, editor, *Advances in Cryptology—CRYPTO '94*, volume 839 of *Lecture Notes in Computer Science*, pages 75–82. Springer-Verlag, 21–25 Aug. 1994.

[20] D. Bleichenbacher and U. M. Maurer. On the efficiency of one-time digital signatures. In K. Kim and T. Matsumoto, editors, *ASIACRYPT*, volume 1163 of *Lecture Notes in Computer Science*, pages 145–158. Springer, 1996. ISBN 3-540-61872-4.

[21] J. Blömer and J.-P. Seifert. On the complexity of computing short linearly independent vectors and short bases in a lattice. In *STOC '99: Proceedings of the thirty-first annual ACM symposium on Theory of computing*, pages 711–720, New York, NY, USA, 1999. ACM. ISBN 1-58113-067-8. doi: http://doi.acm.org/10.1145/301250.301441.

[22] M. Blum. How to prove a theorem so no one else can claim it. In *Proceedings of the International Congress of Mathematicians*, pages 1444–1451, 1987.

[23] M. Blum and S. Kannan. Designing programs that check their work. *21st ACM STOC*, pages 86–97, 1989.

[24] M. Blum and S. Kannan. Designing programs that check their work. *J. ACM*, 42(1):269–291, 1995. ISSN 0004-5411. doi: http://doi.acm.org/10.1145/200836.200880.

[25] M. Blum and S. Micali. How to generate cryptographically strong sequences of pseudo-random bits. *SIAM J. Comput.*, 13(4):850–864, 1984. ISSN 0097-5397. doi: http://dx.doi.org/10.1137/0213053.

[26] M. Blum, M. Luby, and R. Rubinfeld. Self-testing/correcting with applications to numerical problems. *Journal of Computer and System Sciences*, 47(3):549–595, 1993.

[27] A. Bogdanov and L. Trevisan. On worst-case to average-case reductions for np problems. *SIAM Journal on Computing*, 36(4):1119–1159, 2006.

[28] A. Bogdanov and L. Trevisan. Average-case complexity. *CoRR*, 2006.

[29] B. Bollobás. On generalized graphs. *Acta Math. Acad. Sci. Hungar*, 16:447–452, 1965. ISSN 0001-5954.

[30] R. B. Boppana, J. Håstad, and S. Zachos. Does co-NP have short interactive proofs? *Information Processing Letters*, 25(2):127–132, 1987.

[31] G. Brassard. Relativized cryptography. In *Proceedings of the 20th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 383–391. IEEE Computer Society, 1979.

[32] G. Brassard and L. Salvail. Quantum merkle puzzles. In *International Conference on Quantum, Nano and Micro Technologies (ICQNM)*, pages 76–79. IEEE Computer Society, 2008.

[33] G. Brassard, D. Chaum, and C. Crépeau. Minimum disclosure proofs of knowledge. *J. Comput. Syst. Sci.*, 37(2):156–189, Oct. 1988.

[34] J. Buchmann, J. Loho, and J. Zayer. An implementation of the general number field sieve. In *CRYPTO '93*, pages 159–165, New York, NY, USA, 1994. Springer-Verlag New York, Inc. ISBN 0-387-57766-1.

[35] Cachin and Maurer. Unconditional security against memory-bounded adversaries. In *CRYPTO: Proceedings of Crypto*, 1997.

[36] C. Cachin, C. Crépeau, and J. Marcil. Oblivious transfer with a memory-bounded receiver. In *Proceedings of the 39th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 493–502. IEEE Computer Society, 1998.

[37] R. Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *FOCS*, pages 136–145, 2001.

[38] R. Canetti, O. Goldreich, and S. Halevi. The random oracle methodology, revisited. In *Proc. 30th STOC*, pages 209–218. ACM, 1998.

[39] R. Canetti, O. Goldreich, S. Goldwasser, and S. Micali. Resettable zero-knowledge (extended abstract). In *STOC*, pages 235–244, 2000.

[40] N. Chandran, V. Goyal, and A. Sahai. New constructions for UC secure computation using tamper-proof hardware. In *EUROCRYPT*, pages 545–562, 2008.

[41] D. Chaum, C. Crépeau, and I. Damgård. Multiparty unconditionally secure protocols (extended abstract). In *STOC*, pages 11–19, 1988.

[42] C. Crépeau and J. Kilian. Achieving oblivious transfer using weakened security assumptions (extended abstract). In *FOCS*, pages 42–52, 1988.

[43] J. Daemen and V. Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard*. Springer, 2002. ISBN 3-540-42580-2.

[44] I. Damgård. Collision free hash functions and public key signature schemes. In *Advances in Cryptology – EUROCRYPT '87*, volume 304 of *Lecture Notes in Computer Science*, pages 203–216. Springer, 1987.

[45] I. Damgård, O. Goldreich, T. Okamoto, and A. Wigderson. Honest verifier vs. dishonest verifier in public coin zero-knowledge proofs. In *Advances in Cryptology – CRYPTO '95*, volume 963 of *Lecture Notes in Computer Science*, pages 325–338. Springer, 1995.

[46] I. B. Damgård, T. P. Pedersen, and B. Pfitzmann. Statistical secrecy and multibit commitments. *IEEE Transactions on Information Theory*, 44(3):1143–1151, 1998.

[47] W. Diffie and M. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22(6):644–654, Nov. 1976.

[48] Y. Z. Ding, D. Harnik, A. Rosen, and R. Shaltiel. Constant-round oblivious transfer in the bounded storage model. In *Theory of Cryptography, First Theory of Cryptography Conference, TCC 2004*, pages 446–472, 2004.

[49] I. Dinur, G. Kindler, R. Raz, and S. Safra. Approximating cvp to within almost-polynomial factors is np-hard. *Combinatorica*, 23(2):205–243, 2003. ISSN 0209-9683. doi: http://dx.doi.org/10.1007/s00493-003-0019-y.

[50] C. Dwork, U. Feige, J. Kilian, M. Naor, and S. Safra. Low communication 2-prover zero-knowledge proofs for np. In *CRYPTO*, pages 215–227, 1992.

[51] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. pages 10–18, 1984.

[52] S. Even, A. L. Selman, and Y. Yacobi. The complexity of promise problems with applications to public-key cryptography. *Inf. Control*, 61(2):159–173, 1984. ISSN 0019-9958. doi: http://dx.doi.org/10.1016/S0019-9958(84)80056-X.

[53] S. Even, O. Goldreich, and S. Micali. On-line/off-line digital signatures. *J. Cryptology*, 9(1):35–67, 1996. Preliminary version in CRYPTO '89.

[54] J. Feigenbaum and L. Fortnow. Random-self-reducibility of complete sets. *SIAM Journal on Computing*, 22(5):994–1005, 1993.

[55] J. Feigenbaum, L. Fortnow, C. Lund, and D. Spielman. The power of adaptiveness and additional queries in random-self-reductions. *Computational Complexity*, 4:338–346, 1994.

[56] L. Fortnow. The complexity of perfect zero-knowledge. *Advances in Computing Research: Randomness and Computation*, 5:327–343, 1989.

[57] L. Fortnow, J. Rompel, and M. Sipser. On the power of multi-prover interactive protocols. In *Theoretical Computer Science*, pages 156–161, 1988.

[58] L. Fortnow, J. Rompel, and M. Sipser. On the power of multi-prover interactive protocols. *Theoretical Computer Science*, 134(2):545–557, 1994.

[59] R. Gennaro and L. Trevisan. Lower bounds on the efficiency of generic cryptographic constructions. In *Proc. 41st FOCS*, pages 305–313. IEEE, 2000.

[60] R. Gennaro, Y. Gertner, and J. Katz. Lower bounds on the efficiency of encryption and digital signature schemes. In *Proc. 35th STOC*. ACM, 2003.

[61] R. Gennaro, Y. Gertner, J. Katz, and L. Trevisan. Bounds on the efficiency of generic cryptographic constructions. *SICOMP: SIAM Journal on Computing*, 35, 2005. Preliminary versions in FOCS' 00 [59] and STOC' 03 [60].

[62] O. Goldreich. *Foundations of Cryptography: Basic Applications*. Cambridge University Press, 2004.

[63] O. Goldreich. On promise problems (a survey in memory of Shimon Even [1935-2004]). Technical Report TR05–018, Electronic Colloquium on Computational Complexity, February 2005.

[64] O. Goldreich and S. Goldwasser. On the possibility of basing cryptography on the assumption that P $\neq$ NP. Theory of Cryptography Library: Record 98-05, February 1998. `http://theory.lcs.mit.edu/~tcryptol`.

[65] O. Goldreich and S. Goldwasser. On the limits of nonapproximability of lattice problems. *Journal of Computer and System Sciences*, 60(3):540–563, 2000. ISSN 0022-0000. 30th Annual ACM Symposium on Theory of Computing (Dallas, TX, 1998).

[66] O. Goldreich and A. Kahan. How to construct constant-round zero-knowledge proof systems for NP. *Journal of Cryptology*, 9(3):167–189, Summer 1996.

[67] O. Goldreich and R. Ostrovsky. Software protection and simulation on oblivious rams. *J. ACM*, 43(3):431–473, 1996.

[68] O. Goldreich and S. P. Vadhan. Comparing entropies in statistical zero knowl-

edge with applications to the structure of SZK. In *IEEE Conference on Computational Complexity*, pages 54–73. IEEE Computer Society, 1999.

[69] O. Goldreich, Y. Mansour, and M. Sipser. Interactive proof systems: Provers that never fail and random selection. *FOCS*, 0:449–461, 1987. ISSN 0272-5428.

[70] O. Goldreich, S. Micali, and A. Wigderson. How to play ANY mental game. In ACM, editor, *Proc. 19th STOC*, pages 218–229. ACM, 1987. See [62, Chap. 7] for more details.

[71] O. Goldreich, S. Micali, and A. Wigderson. Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems. *J. ACM*, 38 (3):691–729, July 1991. Preliminary version in FOCS' 86.

[72] O. Goldreich, S. Goldwasser, and D. Ron. Property testing and its connection to learning and approximation. *J. ACM*, 45(4):653–750, 1998. ISSN 0004-5411. doi: http://doi.acm.org/10.1145/285055.285060.

[73] O. Goldreich, A. Sahai, and S. Vadhan. Honest-verifier statistical zero-knowledge equals general statistical zero-knowledge. In *In Proceedings of the 30th Annual ACM Symposium on Theory of Computing*, pages 399–408, 1998.

[74] O. Goldreich, S. Vadhan, and A. Wigderson. On interactive proofs with a laconic prover. In *Proc. 28th ICALP*, 2001.

[75] S. Goldwasser and M. Sipser. Private coins versus public coins in interactive proof systems. *Advances in Computing Research: Randomness and Computation*, 5:73–90, 1989.

[76] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interac-

tive proof systems. *SIAM J. Comput.*, 18(1):186–208, 1989. Preliminary version in STOC' 85.

[77] S. Goldwasser, Y. T. Kalai, and G. Rothblum. One-time programs. In *CRYPTO*, Lecture Notes in Computer Science, pages 39–56. Springer, 2008.

[78] S. Goldwasser, Y. T. Kalai, and G. N. Rothblum. Delegating computation: interactive proofs for muggles. In *STOC*, pages 113–122, 2008.

[79] S. D. Gordon, H. Wee, A. Yerukhimovich, and D. Xiao. On the round complexity of zero-knowledge proofs from one-way permutations, 2009. Manuscript. Available at `http://www.cs.princeton.edu/~dxiao/docs/zk-owp.pdf`.

[80] V. Goyal, Y. Ishai, A. Sahai, R. Venkatesan, and A. Wadia. Founding cryptography on tamper-proof hardware tokens. In *TCC*, 2010.

[81] L. K. Grover. A fast quantum mechanical algorithm for database search. In *Annual Symposium on Theory of Computing*, pages 212–219, 22–24 May 1996.

[82] Haitner, Hoch, Reingold, and Segev. Finding collisions in interactive protocols – A tight lower bound on the round complexity of statistically-hiding commitments. In *ECCCTR: Electronic Colloquium on Computational Complexity, technical reports*, 2007.

[83] I. Haitner and O. Reingold. A new interactive hashing theorem. In *Proceedings of the 18th Annual IEEE Conference on Computational Complexity*, 2007. See also preliminary draft of full version, `research.microsoft.com/en-us/um/people/iftach/papers/IneractiveHashing/IneractiveHashing_Jour_1.pdf`.

[84] I. Haitner, O. Reingold, S. Vadhan, and H. Wee. Inaccessible entropy. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing (STOC)*.

[85] I. Haitner, O. Horvitz, J. Katz, C. Koo, R. Morselli, and R. Shaltiel. Reducing complexity assumptions for statistically-hiding commitment. In *Advances in Cryptology – EUROCRYPT 2005*, pages 58–77, 2005. See also preliminary draft of full version, `www.wisdom.weizmann.ac.il/~iftachh/papers/SCfromRegularOWF.pdf`.

[86] I. Haitner, J. J. Hoch, O. Reingold, and G. Segev. Finding collisions in interactive protocols – A tight lower bound on the round complexity of statistically-hiding commitments. In *Proceedings of the 47th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE Computer Society, 2007.

[87] I. Haitner, M. Nguyen, S. J. Ong, O. Reingold, and S. Vadhan. Statistically-hiding commitments and statistical zero-knowledge arguments from any one-way function. To appear in SIAM Journal on Computing, November 2007.

[88] I. Haitner, O. Reingold, S. Vadhan, and H. Wee. Inaccessible entropy. In *STOC '09: Proceedings of the 41st annual ACM symposium on Theory of computing*, 2009.

[89] I. Haitner, A. Rosen, and R. Shaltiel. On the (im)possibility of arthur-merlin witness hiding protocols. In *Theory of Cryptography, Fourth Theory of Cryptography Conference, TCC 2009*, 2009.

[90] I. Haitner, O. Reingold, and S. Vadhan. Improvements in constructions of pseudorandom generators from one-way functions. In *Proceedings of the 42nd Annual ACM Symposium on Theory of Computing (STOC)*, 2010.

[91] Y. Han, L. A. Hemaspaandra, and T. Thierauf. Threshold computation and cryptographic security. *SIAM J. Comput.*, 26(1):59–78, 1997. ISSN 0097-5397.

[92] J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999. Preliminary versions appeared in STOC' 89 and STOC' 90.

[93] C. Hazay and Y. Lindell. Constructions of truly practical secure protocols using standardsmartcards. In *ACM Conference on Computer and Communications Security*, pages 491–500, 2008.

[94] E. Hemaspaandra, A. V. Naik, M. Ogihara, and A. L. Selman. P-selective sets and reducing search to decision vs. self-reducibility. *J. Comput. Syst. Sci.*, 53(2): 194–209, 1996. ISSN 0022-0000. doi: http://dx.doi.org/10.1006/jcss.1996.0061.

[95] T. Holenstein. Pseudorandom generators from one-way functions: A simple construction for any hardness. In *Theory of Cryptography, Third Theory of Cryptography Conference, TCC 2006*, 2006.

[96] T. Holenstein. Private communication. 2009.

[97] R. Impagliazzo and M. Luby. One-way functions are essential for complexity based cryptography (extended abstract). In *Proc.* 30*th FOCS*, pages 230–235. IEEE, 1989.

[98] R. Impagliazzo and S. Rudich. Limits on the provable consequences of one-way permutations. In *Proc.* 21*st STOC* sto [1], pages 44–61. Full version available from Russell Impagliazzo's home page.

[99] R. Impagliazzo and A. Wigderson. $\mathbf{P} = \mathbf{BPP}$ if $\mathbf{E}$ requires exponential circuits:

Derandomizing the XOR lemma. In *Proc. 29th STOC*, pages 220–229. ACM, 1997.

[100] Y. Ishai, E. Kushilevitz, and R. Ostrovsky. Sufficient conditions for collision-resistant hashing. In *In Proceedings of the 2nd Theory of Cryptography Conference*, pages 445–456, 2005.

[101] Y. Ishai, M. Prabhakaran, and A. Sahai. Founding cryptography on oblivious transfer - efficiently. In D. Wagner, editor, *CRYPTO*, volume 5157 of *Lecture Notes in Computer Science*, pages 572–591. Springer, 2008. ISBN 978-3-540-85173-8.

[102] Y. T. Kalai and R. Raz. Interactive PCP. In *ICALP (2)*, pages 536–547, 2008.

[103] L. V. Kantorovich. On the translocation of masses. *Doklady Akademii Nauk SSSR*, 37:227–229, 1942.

[104] L. V. Kantorovich and G. S. Rubinstein. On a space of totally additive functions. *Vestn Lening. Univ*, 13(7):52–59, 1958.

[105] J. Katz. Universally composable multi-party computation using tamper-proof hardware. In *EUROCRYPT*, Lecture Notes in Computer Science, pages 115–128. Springer, 2007.

[106] S. Khot. Hardness of approximating the shortest vector problem in lattices. *J. ACM*, 52(5):789–808, 2005. ISSN 0004-5411. doi: http://doi.acm.org/10.1145/1089023.1089027.

[107] J. Kilian. Founding cryptography on oblivious transfer. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing (STOC)*, pages 20–31, 1988.

[108] J. Kilian, E. Petrank, and G. Tardos. Probabilistically checkable proofs with zero knowledge. In *STOC: ACM Symposium on Theory of Computing (STOC)*, 1997.

[109] J. H. Kim, D. R. Simon, and P. Tetali. Limits on the efficiency of one-way permutation-based hash functions. In *FOCS*, pages 535–542, 1999. URL `http://computer.org/proceedings/focs/0409/04090535abs.htm`.

[110] V. Kolesnikov. Truly efficient string oblivious transfer using resettable tamper-proof tokens. In *TCC*, pages 327–342, 2010.

[111] Kushilevitz, Lindell, and Rabin. Information-theoretically secure protocols and security under composition. In *STOC: ACM Symposium on Theory of Computing (STOC)*, 2006.

[112] L. Lamport. Constructing digital signatures from a one-way function. Technical Report CSL-98, SRI International, Oct. 1979.

[113] D. Lapidot and A. Shamir. A one-round, two-prover, zero-knowledge protocol for np. *Combinatorica*, 15(2):204–214, 1995.

[114] Y. Lindell. Parallel coin-tossing and constant-round secure two-party computation. *JCRYPTOLOGY*, 2003.

[115] C. Lund, L. Fortnow, H. Karloff, and N. Nisan. Algebraic methods for interactive proof systems. In *Proc. 31st FOCS*, pages 2–10. IEEE, 1990.

[116] U. M. Maurer. Conditionally-perfect secrecy and a provably-secure randomized cipher. *J. Cryptology*, 5(1):53–66, 1992.

[117] N. Megiddo and C. H. Papadimitriou. On total functions, existence theorems and computational complexity. *Theor. Comput. Sci.*, 81(2):317–324, 1991. ISSN 0304-3975. doi: http://dx.doi.org/10.1016/0304-3975(91)90200-L.

[118] R. Merkle. Secure communications over insecure channels. *Communications of the ACM*, 21(4):294–299, 1978.

[119] R. C. Merkle. A digital signature based on a conventional encryption function. In C. Pomerance, editor, *Advances in Cryptology—CRYPTO '87*, volume 293 of *Lecture Notes in Computer Science*, pages 369–378. Springer-Verlag, 1988, 16–20 Aug. 1987.

[120] D. Micciancio. Lattice based cryptography. In H. C. A. van Tilborg, editor, *Encyclopedia of Cryptography and Security*. 2005. ISBN 978-0-387-23473-1.

[121] D. Micciancio and O. Regev. Worst-case to average-case reductions based on gaussian measures. In *FOCS '04*, pages 372–381, Washington, DC, USA, 2004. IEEE Computer Society. ISBN 0-7695-2228-9. doi: http://dx.doi.org/10.1109/FOCS.2004.72.

[122] G. Monge. Mmoire sur la thorie des dblais et des remblais. *Histoire de l'Acadmie des Sciences de Paris*, page 666, 1781.

[123] T. Moran and G. Segev. David and Goliath commitments: UC computation for asymmetric parties using tamper-proof hardware. In *EUROCRYPT*, pages 527–544, 2008.

[124] M. Naor and M. Yung. Universal one-way hash functions and their cryptographic applications. In *Proc. 21st STOC* sto [1], pages 33–43.

[125] M. Naor, R. Ostrovsky, R. Venkatesan, and M. Yung. Perfect zero-knowledge arguments for NP using any one-way permutation. *Journal of Cryptology*, 11 (2):87–108, 1998. Preliminary version in *CRYPTO'92*.

[126] National Institute of Standards and Technology. *FIPS PUB 180-1: Secure Hash Standard*. National Institute for Standards and Technology, Apr. 1995.

[127] N. Nisan and A. Wigderson. Hardness vs randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, Oct. 1994. Preliminary version in FOCS' 88.

[128] R. O'Donnell. Hardness amplification within np. *J. Comput. Syst. Sci.*, 69(1): 68–94, 2004. ISSN 0022-0000. doi: http://dx.doi.org/10.1016/j.jcss.2004.01.001.

[129] T. Okamoto. On relationships between statistical zero-knowledge proofs. *Journal of Computer and System Sciences*, 60(1):47–108, 2000. Preliminary version in *STOC'96*.

[130] S. J. Ong and S. P. Vadhan. An equivalence between zero knowledge and commitments. In *TCC*, pages 482–500, 2008.

[131] R. Ostrovsky and A. Wigderson. One-way functions are essential for non-trivial zero-knowledge. Technical Report TR-93-073, International Computer Science Institute, Berkeley, CA, Nov. 1993. Preliminary version in Proc. 2nd Israeli Symp. on Theory of Computing and Systems, 1993, pp. 3–17.

[132] R. Ostrovsky, R. Venkatesan, and M. Yung. Interactive hashing simplifies zero-knowledge protocol design. pages 267–273, 1993.

[133] R. Ostrovsky, R. Venkatesan, and M. Yung. Fair games against an all-powerful

adversary. *AMS DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 155–169, 1993. Preliminary version in *SEQUENCES'91*.

[134] C. H. Papadimitriou. On the complexity of the parity argument and other inefficient proofs of existence. *J. Comput. Syst. Sci.*, 48(3):498–532, 1994. ISSN 0022-0000. doi: http://dx.doi.org/10.1016/S0022-0000(05)80063-7.

[135] R. Pass. Parallel repetition of zero-knowledge proofs and the possibility of basing cryptography on np-hardness. In *IEEE Conference on Computational Complexity*, pages 96–110, 2006.

[136] A. Pavan, A. L. Selman, S. Sengupta, and N. V. Vinodchandran. Polylogarithmic-round interactive proofs for conp collapse the exponential hierarchy. *Theor. Comput. Sci.*, 385(1-3):167–178, 2007. ISSN 0304-3975. doi: http://dx.doi.org/10.1016/j.tcs.2007.06.013.

[137] C. Peikert and V. Vaikuntanathan. Noninteractive statistical zero-knowledge proofs for lattice problems. In *CRYPTO 2008: Proceedings of the 28th Annual conference on Cryptology*, pages 536–553, Berlin, Heidelberg, 2008. Springer-Verlag. ISBN 978-3-540-85173-8. doi: http://dx.doi.org/10.1007/978-3-540-85174-5_30.

[138] A. Perrig, R. Canetti, D. Song, and J. D. Tygar. Efficient authentication and signing of multicast streams over lossy channels. In *IEEE Security and Privacy Symposium*, 2000.

[139] M. Rabin. How to exchange secrets by oblivious transfer. Technical Report TR-81, Harvard Aiken Computation Laboratory, 1981.

[140] M. O. Rabin. Digitalized signatures. In R. A. DeMillo, D. P. Dobkin, A. K.

Jones, and R. J. Lipton, editors, *Foundations of Secure Computation*, pages 155–168. Academic Press, 1978.

[141] M. O. Rabin. Digitalized signatures and public-key functions as intractable as factorization. Technical Report MIT/LCS/TR-212, Massachusetts Institute of Technology, Jan. 1979. URL `ftp://ftp-pubs.lcs.mit.edu/pub/lcs-pubs/tr.outbox/MIT-LCS-TR-212.ps.gz`.

[142] T. Rabin and M. Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority (extended abstract). In *STOC*, pages 73–85, 1989.

[143] O. Reingold, L. Trevisan, and S. P. Vadhan. Notions of reducibility between cryptographic primitives. In *Theory of Cryptography, First Theory of Cryptography Conference, TCC 2004*, volume 2951 of *Lecture Notes in Computer Science*, pages 1–20. Springer, 2004.

[144] R. L. Rivest, A. Shamir, and L. M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2): 120–126, Feb 1978.

[145] Y. Rubner, C. Tomasi, and L. J. Guibas. A metric for distributions with applications to image databases. In *ICCV '98: Proceedings of the Sixth International Conference on Computer Vision*, 1998.

[146] A. Sahai and S. Vadhan. A complete problem for statistical zero knowledge. *Journal of the ACM*, 50(2):196–249, 2003. Preliminary version in *FOCS'97*.

[147] S. Sanghvi and S. P. Vadhan. The round complexity of two-party random selection. In *STOC*, pages 338–347, 2005.

[148] A. Shamir. IP = PSPACE. *Journal of the ACM*, 39(4):869–877, 1992.

[149] P. W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26:1484–1509, 1997.

[150] D. Simon. Finding collisions on a one-way street: Can secure hash functions be based on general assumptions? In *Advances in Cryptology – EUROCRYPT '98*, volume 1403 of *Lecture Notes in Computer Science*, pages 334–345. Springer, 1998.

[151] Trevisan. On uniform amplification of hardness in NP. In *STOC: ACM Symposium on Theory of Computing (STOC)*, 2005.

[152] S. Vadhan. Private communication., 2009.

[153] S. P. Vadhan. *A Study of Statistical Zero-Knowledge Proofs*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 1999.

[154] L. G. Valiant and V. V. Vazirani. Np is as easy as detecting unique solutions. In *STOC '85: Proceedings of the seventeenth annual ACM symposium on Theory of computing*, 1985.

[155] S. Vaudenay. One-time identification with low memory. In *Eurocode 92*, number 339 in CISM Courses and Lectures, pages 217–228, Wien, 1992. Springer-Verlag, Berlin Germany.

[156] H. Wee. One-way permutations, interactive hashing and statistically hiding commitments. In S. P. Vadhan, editor, *TCC*, volume 4392 of *Lecture Notes in Computer Science*, pages 419–433. Springer, 2007. ISBN 3-540-70935-5. URL `http://dx.doi.org/10.1007/978-3-540-70936-7_23`.

[157] D. Xiao. (Nearly) optimal black-box constructions of commitments secure against selective opening attacks, 2009. Manuscript.

[158] A. C. Yao. Theory and applications of trapdoor functions. In *Proc.* 23*rd FOCS*, pages 80–91. IEEE, 1982.

[159] C.-K. Yap. Some consequences of non-uniform conditions on uniform classes. *Theor. Comput. Sci.*, 26:287–300, 1983.