New Perspectives on the Complexity of Computational Learning, and Other Problems in Theoretical Computer Science

David Xiao

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
Adviser: Boaz Barak

September 2009

Abstract

In this thesis we present the following results.

- Learning theory, and in particular PAC learning, was introduced by Valiant [CACM 1984] and has since become a major area of research in theoretical and applied computer science. One natural question that was posed at the very inception of the field is whether there are classes of functions that are hard to learn.

  PAC learning is hard under widely held conjectures such as the existence of one-way functions, and on the other hand it is known that if PAC learning is hard then $\mathbf{P} \neq \mathbf{NP}$. We further study sufficient and necessary conditions for PAC learning to be hard, and we prove that:

  1. $\mathbf{ZK} \neq \mathbf{BPP}$ implies that PAC learning is hard.
  2. It is unlikely using standard techniques that one can prove that PAC learning is hard implies that $\mathbf{ZK} \neq \mathbf{BPP}$.
  3. It is unlikely using standard techniques that one can prove that $\mathbf{P} \neq \mathbf{NP}$ implies that $\mathbf{ZK} \neq \mathbf{BPP}$.

  Here, "standard techniques" refers to various classes of efficient reductions. Together, these results imply that the hardness of PAC learning lies between the non-triviality of **ZK** on the one hand and the hardness of **NP** on the other hand. Furthermore, the hardness of PAC learning lies "strictly" between the two, in the sense that most standard techniques cannot prove equivalence with either $\mathbf{ZK} \neq \mathbf{BPP}$ or $\mathbf{NP} \neq \mathbf{P}$.

  In proving these results, we show new connections between PAC learning and auxiliary-input one-way functions, which were defined by Ostrovsky and Wigder-

son [ISTCS 1993] to better understand **ZK**. We also define new problems related to PAC learning that we believe of are independent interest, and may be useful in future studies of the complexity of PAC learning.

- A secure *failure-localization* (FL) protocol allows a sender to localize faulty links on a single path through a network to a receiver, even when intermediate nodes on the path behave adversarially. Such protocols were proposed as tools that enable Internet service providers to select high-performance paths through the Internet, or to enforce contractual obligations. We give the first formal definitions of security for FL protocols and prove that for such protocols, security requires *each* intermediate node on the path to have some shared secret information (*e.g.* keys), and that every *black-box* construction of a secure FL protocol from a random oracle requires *each* intermediate node to invoke the random oracle. This suggests that achieving this kind of security is unrealistic as intermediate nodes have little incentive to participate in the real world.

- Ahlswede and Winter [IEEE Trans. Inf. Th. 2002] introduced a Chernoff bound for matrix-valued random variables, which is a non-trivial generalization of the usual Chernoff bound for real-valued random variables. We present an efficient derandomization of their bound using the method of pessimistic estimators (see Raghavan [JCSS 1988]). As a consequence, we derandomize a construction of Alon and Roichman [RSA 1994] to efficiently construct an expanding Cayley graph of logarithmic degree on any (possibly non-abelian) group. This gives an optimal solution to the homomorphism testing problem of Shpilka and Wigderson [STOC 2004]. We also apply these pessimistic estimators to the problem of solving semi-definite covering problems, thus giving a deterministic algorithm for the quantum hypergraph cover problem of Ahslwede and Winter.

Acknowledgements

Throughout the course of my Ph D I've had the privilege of working with some of the most outstanding and wonderful researchers in theoretical computer science. I feel most grateful for the honor of being advised by Boaz Barak and Avi Wigderson. One of the first things Boaz taught me was that the best way to learn about an area is to try and solve the open problems in that area, and I am indebted for the pro-active attitude towards research that he has imparted on me. He certainly led by example, and would not hesitate to spend 8 hours straight working on a problem if that was necessary (luckily it was only necessary once!).

Conversations with Avi were always a pleasure and sometimes we would arrive at the end of a meeting and realize that we hadn't even started talking about research! Fortunately that only happened once in a while, otherwise I would have missed out on all the things that he taught me about complexity, about research, and about being a scientist. There were days when I would arrive at meetings discouraged about not making any progress, but his enthusiasm for research and the excitement he brought to each discussion was infectious, and I would walk away with new ideas and new optimism. By not only advancing the state of the art in our field but also taking the time to explain our area to other mathematicians, other scientists, or even laypeople, Avi has also taught me that effectively communicating great ideas is just as important as discovering them.

The list of people who have helped and encouraged me throughout these last few years is long and unfortunately I will surely leave some out inadvertently in these acknowledgements. Salil Vadhan, who guided my undergraduate research, has remained a valuable colleague who has always had insightful comments and suggestions for the research questions I've asked him. I was fortunate that Benny Applebaum was

a postdoc at Princeton during my Ph D, and our research together not only directly produced several of the results in this thesis, but also inspired the questions that led to other results in this work. Sharon Goldberg and I left Princeton to move to New York at the same time, and this led to many long hours sitting together at Columbia's libraries, where luckily once in a while we found a few minutes to take a break from gossiping and actually do some work. I greatly enjoyed the time spent brainstorming with Mohammad Mahmoody-Ghidary, and apologize to the other theory students that we might have annoyed with our heated discussions. Thanks to Barbara Terhal and IBM Research for one lovely summer, and to Andrej Bogdanov and Tsinghua University for another lovely summer. Thanks to Hoeteck Wee and Luca Trevisan, you guys made conferences and workshops much more fun. Thanks to my committee, Sanjeev Arora, Russell Impagliazzo, and Rob Schapire. And, in no particular, thanks to Ronen Shaltiel, Tal Malkin, Eran Tromer, Jennifer Rexford, and Iftach Haitner for many insightful discussions and the pleasure of working with them.

I was supported in my research by an NSF Graduate Research Fellowship, an NDSEG Fellowship, a Princeton University Upton Fellowship, and in part by NSF grants CNS-0627526, CCF-0426582 and CCF-0832797, and a grant from the Packard foundation

Thanks to Damian Carrieri, Patrick Bradley, and Eve Schneider for letting me crash on their couch more times than is socially acceptable. Thanks to Nathan Ha for navigating the dark corners of New York with me. And much love to my urban family in New York, Tacara Soones and Jiajia Ye, life here for the last three years would have been impossible without you. I'll miss our cooking sessions and our evenings spent over-analyzing each others' lives, but I know our commitment to friendship will endure even though our time in New York is over. Best of luck on the West Coast and wherever the currents take you.

爸爸妈妈，最需要感谢的是你们。我从小到大所作成的一切都是由于你们的支持、爱情、关心。你们一直培养了我的好奇心，不管什么书都愿意买，一本不够就买两本，两本不够就买三本，想学中文就送我到中文学校，想学计算机就买最新最快的一台。

小时候说弹钢琴是为了学本事，上学念书也是的，这样才可以创造更美好的未来。这么多年的书现在念完了，下来也没有书好念了。上课读书确实学到了不少东西，可是一个人生活中最难学的不是弹钢琴，也不是计算机，而是如何作一个好人。这一个最难学的本事我是从你们学来的。世界上没有很多像你们这样的父母亲，有了你们我非常感激。

Contents

Chapter 1

Introduction and Preliminaries

Theoretical computer science is a young but incredibly broad field. Despite the diversity of topics studied under this umbrella, there are several unifying concepts and techniques that underlie much of the science. One such recurring theme is the use of *reductions*, perhaps *the* central proof technique in theoretical computer science. The intuitive notion of a reduction is something even a child could understand (*e.g.* in order to save the princess, it suffices to slay the dragon). In computational complexity we always require that the reduction be *efficient*, namely it must run in polynomial time. In principle, besides this sole efficiency requirement the reduction can be as creative or as bizarre as it likes.

In practice however, most of the reductions that we build are much more constrained, *i.e.* they are relativizing, or black-box, or some variant thereof. Starting with work by Baker *et al.* [BGS75], computer scientists have studied whether such constrained reductions can resolve open questions, for example **P** vs **NP**. In their groundbreaking work, [BGS75] proved that relativizing techniques are insufficient to resolve the **P** vs **NP** question, thereby ruling out techniques such as diagonalization, which people had

previously hoped to apply to the problem. By showing that relativizing techniques are insufficient to address the **P** vs **NP** problem, [BGS75] provides insight into the difficulty of these questions, as well as indications of the obstacles that need to be overcome in order to answer them.

In this thesis, we continue to explore the limits of various kinds of reductions, and in particular we apply this methodology to the complexity of computational learning. Computational learning was introduced by Valiant [Val84] to model algorithms that are supposed to *efficiently* learn from labeled distributions. Since then, it has been one of the most important and widely studied areas within theoretical computer science, and therefore understanding its complexity is invaluable.

In Chapter 2 through Chapter 5, we explore the complexity of PAC learning and relate it to the non-triviality of zero knowledge and the hardness of **NP**. We refine the known sufficient and necessary conditions for PAC learning to be hard by showing that not only does the existence of one-way functions imply that PAC learning is hard, but so does the weaker assumption that $\mathbf{ZK} \neq \mathbf{BPP}$. We then explore what kinds of *reductions* may be useful to prove equivalence of **NP**-hardness and the hardness of PAC learning, or equivalence of the non-triviality of **ZK** and the hardness of PAC learning. A more detailed overview of these results may be found in Chapter 2.

In Chapter 6 we also apply the methodology of studying reductions to a security problem in network routing called failure localization. In this setting, messages from a sender to a receiver must be sent through a series of intermediate, untrusted nodes. We show that security in this setting requires that all the intermediate nodes must actively participate in the protocol by both maintaining a key infrastructure, as well as performing cryptographic computations. Our result is proven by showing that a black-box reduction that constructs such a scheme from a random oracle (or a one-

way function) cannot be secure unless the intermediate nodes actively participate in the scheme.

A second major recurring theme in theoretical computer science is the use of randomness as a valuable computational resource. There are examples of problems where random coin flips enabled us to efficiently perform tasks that otherwise seem intractable (*e.g.* polynomial identity testing), or whose deterministic polynomial-time algorithms are impractical (*e.g.* primality testing, [Mil75, SS77, Rab80]). However, in a series of breakthrough works [Yao82, BM84, NW88, IW97, IW98], it was shown that if plausible hardness assumptions hold, then in fact randomness does not give any superpolynomial speedup over deterministic computation. Thus, the field of *derandomization* was born, which is concerned with reducing or eliminating the need for random coins from algorithms. Although we have no hope using current techniques of unconditionally proving that $\mathbf{P} = \mathbf{BPP}$, nevertheless we can unconditionally derandomize certain specific algorithms, and this has led to breakthrough works (*e.g.* in primality testing, [AKS02]).

In Chapter 7 we will show how to unconditionally derandomize a probabilistic inequality due to Ahlswede and Winter [AW02] that generalizes the classical Chernoff bound to the case of random variables that take values in the space of positive semi-definite matrices. This leads to several applications in computer science, most notably in giving an efficient deterministic construction of $O(\log n)$-degree Cayley expanders for arbitrary groups, as well as to a way to derandomize the rounding procedures for semi-definite programs solving quantum hypergraph covering problems.

More detailed introductions into each of these topics can be found in their respective chapters. In the remainder of this chapter, we present some basic notation and definitions that will be used throughout this thesis, as well as some background results

that will be useful to us.

1.1 Basic notation

We say a function $\varepsilon(n)$ is negligible (with respect to n) if for all $c > 0$, it holds that $\varepsilon(n) < n^{-c}$ for all n large enough. Similarly, ε is non-negligible if there exists a $c > 0$ such that $\varepsilon(n) \geq n^{-c}$ for all n large enough.

We will typically let U_n denote the uniform distribution over $\{0,1\}^n$. For two distributions X, Y over a common universe S, we let $\Delta(X,Y)$ denote their statistical distance:

$$\Delta(X,Y) = \frac{1}{2}\sum_{s\in S} |\Pr[X = s] - \Pr[Y = s]|$$

Equivalently, if we look at a distribution X as a vector in $\mathbb{R}^{|S|}$ with non-negative coordinates and whose entries sum to 1, then $\Delta(X,Y) = \frac{1}{2}|X - Y|_1$ the ℓ_1 norm. From this definition, it is clear that statistical distance obeys the triangle inequality, *i.e.* for all distributions Z,

$$\Delta(X,Y) \leq \Delta(X,Z) + \Delta(Y,Z)$$

It is well-known that this is equivalent to the maximal distinguishing probability between the distributions over all statistical tests, namely:

$$\Delta(X,Y) = \max_{T\subseteq S} |\Pr[X \in T] - \Pr[Y \in T]|$$

We say that two families of distributions $\{X_n\}, \{Y_n\}$ over a family of universes $\{S_n\}$ are statistically indistinguishable if $\Delta(X_n, Y_n) \leq \varepsilon(n)$ where ε is negligible in n.

We say that two families of distributions $\{X_n\}, \{Y_n\}$ over a family of universes $\{S_n\}$ are computationally indistinguishable if for all families of circuits $C_n : S_n \to \{0,1\}$

of size poly(n), it holds that $|\Pr[C_n(X_n) = 1] - \Pr[C_n(Y_n) = 1]| \leq \varepsilon(n)$ where ε is a negligible function of n.

For a function $f : \{0,1\}^n \to \{0,1\}^m$ and any $y \in \{0,1\}^m$, let $f^{-1}(y) = \{x \mid f(x) = y\}$. For a distribution X, let $f(X)$ denote the induced output distribution, namely where the probability of y is $\Pr[f(X) = y]$.

We say that a circuit $C : \{0,1\}^n \to \{0,1\}^m$ samples a distribution X if the distribution $C(U_n)$ is identical to X. We say that a distribution X over $\{0,1\}^m$ is efficiently samplable if there exists a circuit C of size poly(m) such that $C(U_n) = X$. The number of circuits of size s is bounded by $2^{O(s \log s)}$, and the same holds for circuits allowed oracle gates.

1.2 Complexity

Throughout this thesis, "efficient" or "efficiency" will always refer to running in polynomial time. The term "algorithm" refers to uniform computation unless explicitly noted otherwise, while the terms "circuit" and "family of circuits" refer to non-uniform computation. Unless otherwise specified, "algorithm" and "circuit" usually refer to *efficient* computations. We use "procedure" to refer to algorithms or circuits that are *possibly computationally unbounded*. Our theorems and lemmas will often specify whether they hold with respect to uniform or non-uniform models of computation; no mention of uniformity means that the result holds for both uniform and non-uniform models.

For a size function $s(n)$, we let $\mathsf{SIZE}(s(n))$ denote the class of functions computable by a non-uniform family of circuits of size $s(n)$. We let $\mathsf{SIZE}^{\mathcal{O}}(s(n))$ denote the class of oracle circuits that are also allowed $\mathcal{O}$ gates.

An algorithm or circuit is randomized if, in addition to its input, it gets as additional input a random string ω of polynomial length that is drawn uniformly at random.

We assume the reader is familiar with the following standard complexity classes:

1. **P**: the class of languages accepted by deterministic Turing machines running in polynomial time.

2. **P**/*poly*: the class of languages accepted by deterministic Turing machine running in polynomial time with poly(n) bits of advice. Equivalently, the class of languages accepted by families of polynomial-size circuits.

3. **NP**: the class of languages accepted by non-deterministic Turing machines running in polynomial time.

4. **coNP**: the class of languages whose complements are in **NP**.

5. **BPP**: the class of languages accepted by randomized Turing machines with two-sided error.

6. **AM**: the class of languages accepted by 2-message public-coin interactive proofs. Equivalently, the class of languages accepted by $O(1)$-message public-coin interactive proofs.

7. **coAM**: the class of languages whose complements are in **AM**.

8. **PH**: the class of languages accepted by alternating machines running in polynomial times, but with only a constant number of alternations.

9. **PSPACE**: the class of languages accepted by deterministic Turing machines running in polynomial space.

The reader is invited to refer to [AB09] for more detailed definitions. Here we recall some of the standard results and conjectures about these classes.

Conjecture 1.2.1. PH *does not collapse. In particular,* $\mathbf{P} \neq \mathbf{NP}$.

Theorem 1.2.2 ([BHZ87])**.** *If* $\mathbf{coNP} \subseteq \mathbf{AM}$*, then the* **PH** *collapses to the second level.*

We will use a relativized version of the **PSPACE**-complete problem "Quantified Boolean Formula" (QBF), so we recall some standard results about the standard (unrelativized) problem.

Definition 1.2.3. QBF is the language of all formulas of the form

$$\exists x_1, \forall x_2, \exists x_3, \ldots, Q_{n-1} x_{n-1}, Q_n x_n, \varphi(x_1, \ldots, x_n)$$

where x_i are boolean variables and φ is a polynomial-size boolean circuit on n variables.

Theorem 1.2.4. QBF *is complete for* **PSPACE**. *Furthermore, any* QBF *formula can be decided in space* $O(n^2)$ *(and hence time* $2^{O(n^2)}$*).*

We will frequently work with *promise problems* [ESY84] instead of languages when they are more convenient. A promise problem Π is a pair of disjoint sets $\Pi_Y, \Pi_N \subseteq \{0,1\}^*$ (known as the YES instances and NO instances, respectively). Of course, a language is just a special case of a promise problem where $\Pi_Y = L$ and $\Pi_N = \{0,1\}^* \setminus L$.

1.3 Reductions: black-box, relativizing, and otherwise

Reductions are a central technique in complexity and cryptography. At a high level, reductions are a systematic way of using a method that solves one problem in order to solve another problem. Perhaps the most familiar kind of reductions are *Karp* (also called many-to-one) reductions, which were used in the celebrated early results on **NP**-hardness [Kar72].

Definition 1.3.1. A Karp reduction from a promise problem Π^1 to another promise problem Π^2 is an efficient algorithm that maps an instance x of Π^1 into an instance y in Π^2. The reduction should satisfy the following correctness condition: $x \in \Pi^1_Y$ if and only if $y \in \Pi^2_Y$, and similarly $x \in \Pi^1_N$ if and only if $y \in \Pi^2_N$.

A Karp reduction may also be randomized, in which case we require that if $x \in \Pi^1_Y$ then $y \in \Pi^2_Y$ with probability at least $2/3$ and if $x \in \Pi^1_N$ then $y \in \Pi^2_N$.

Classifying reduction techniques is a valuable way to understand how to attack long-standing questions in theoretical computer science. Starting with the work of Baker *et al.* in complexity theory [BGS75] and Impagliazzo and Rudich in cryptography [IR89], studying various subclasses of reductions has provided valuable knowledge and understanding about the hardness of various problems.

A reduction is *relativizing* if it remains valid in the presence of any oracle: the reduction holds even if one augments the model of computation with a function $\mathcal{O}$ that all parties are allowed to call for unit cost. What this means precisely can vary with the context and the problems we are reducing between, so after each main definition (*e.g.* PAC learning, zero knowledge, etc.) we will specify later exactly what relativizing means for that context.

An algorithm A accesses another algorithm B in a *black-box* manner if A only feeds B inputs and then uses its outputs. In particular, if A accesses B in a black-box way then A does not care how B is implemented and does not see the code of B.

Notice that a Karp reduction is black-box in one sense: it can use *any* oracle $\mathcal{O}$ that solves Π^2 in order to to solve Π^1. How $\mathcal{O}$ is implemented is immaterial, as long as for each input y the oracle outputs correctly $\mathcal{O}(y) = 1$ if $y \in \Pi^2_Y$ and outputs $\mathcal{O}(y) = 0$ if $y \in \Pi^2_N$.

We formalize this notion of "black-box" and will study its limits with regard to learning. This class of reductions includes reductions from a promise problem to a more general computational task. Given a formal definition of some task $\mathcal{T}$ (such as PAC learning Definition 2.2.2) one can consider reductions that use an oracle solving $\mathcal{T}$ in order to decide a promise problem Π. Viewed from the contrapositive (which is how we will typically interpret such reductions), this means that such a reduction bases the hardness of $\mathcal{T}$ on the hardness of deciding Π.

Definition 1.3.2. A black-box reduction R that bases the hardness of a computational task $\mathcal{T}$ on the hardness of deciding a promise problem Π is an efficient randomized oracle algorithm that, given any oracle $\mathcal{O}$ that solves the task $\mathcal{T}$, satisfies $\Pr[R^{\mathcal{O}}(x) = 1] \geq 1 - 2^{-n}$ for all $x \in \Pi_Y$ and $\Pr[R^{\mathcal{O}}(x) = 1] \leq 2^{-n}$ for all $x \in \Pi_N$.

Cryptographic reductions typically consist of *two* components: a construction and a security analysis, which are both efficient algorithms. Let us take as an example the task of constructing pseudorandom generators from one-way functions [HILL89]. The *construction* of the pseudorandom function is an efficient algorithm that evaluates the pseudorandom generator given access to a one-way function. The *security analysis* is an efficient algorithm that inverts the underlying one-way function given access to a distinguisher that can distinguish the pseudorandom function from a truly random

function.

For the sake of readability, we will be slightly informal in defining cryptographic notions of reducibility; the reader is encouraged to refer to the excellent work of Reingold *et al.* [RTV04] that carefully considers different notions of reducibility. A cryptographic primitive P is a pair of constraints that define *correctness* and *security*: correctness is typically a syntactic guarantee (for example, a pseudorandom generator must be length-increasing), while security is a typically guarantee that no efficient algorithm or circuit can break some property of a function (for example, no efficient algorithm or circuit can distinguish the output of a pseudorandom generator from truly random).

A cryptographic reduction that uses a primitive P to build a primitive Q takes any function f (we consider only primitives that are functions in this work) that satisfies the correctness condition of P and produces a function g that satisfies the correctness condition of Q. Furthermore, the reduction guarantees that if there exists an adversary breaking the security of g, then there exists an adversary breaking the security of f.

Definition 1.3.3. 1. There exists a relativizing reduction that uses a primitive P to build a primitive Q if for every oracle $\mathcal{O}$, if there P exists relative to $\mathcal{O}$ then Q also exists relative to $\mathcal{O}$.

2. There exists a fully-black-box reduction that uses a primitive P to build a primitive Q if there exists an efficient construction algorithm R_{cons} and efficient security analysis algorithm R_{sec} such that if f satisfies the correctness condition of P then R^f_{cons} satisfies the correctness condition of Q, and for any adversary A that breaks the security of R^f_{cons}, the algorithm R^A_{sec} breaks the security of f.

3. There exists a construction-black-box reduction that uses a primitive P to build

a primitive Q if there exists an efficient construction algorithm R_{cons} such that if f satisfies the correctness condition of P then R^f_{cons} satisfies the correctness condition of Q, and if there exists an efficient oracle adversary A^f that breaks the security of R^f_{cons}, then there exists an efficient oracle algorithm S^f that breaks the security of f.

4. There exists a $\forall\exists$ construction-black-box that uses a primitive P to build a primitive Q if for every f that satisfies the correctness condition of P, there exists an oracle algorithm R_{cons} such that R^f_{cons} satisfies the correctness condition of Q, and if there exists an efficient oracle adversary A^f that breaks the security of R^f_{cons}, then there exists an efficient oracle algorithm S^f that breaks the security of f.

The following relationships between the different kinds of reductions is straightforward from the definition.

Proposition 1.3.4. *1. If a reduction is fully-black-box then it also relativizing.*

2. If a reduction is construction-black-box then it is also $\forall\exists$ construction-black-box.

3. If a reduction is relativizing then it is also $\forall\exists$ construction-black-box.

Efficient-oracle reductions

Efficient-oracle reductions are not black-box *per se*, but only make very limited use of the code of the algorithm implementing the oracle. Namely, such a reduction uses input-output access to the algorithm, but the correctness of the reduction only holds when the oracle can be implemented by a polynomial-time algorithm (or circuit); that is, the reduction is not guaranteed to be correct if given access to an oracle that solves the desired problem but runs in super-polynomial time. As a concrete example,

see the section on the Ostrovsky-Wigderson theorem Section 4.2.1. Cryptographic reductions can be efficient-oracle in the construction or the security analysis.

Adaptivity

Black-box or efficient-oracle reductions can use the oracle in creative ways; one way they may take advantage of the oracle is to use the oracle's answers to generate new queries. This is characterized by the *adaptivity* of the reduction.

Definition 1.3.5. A black-box or efficient-oracle reduction R that uses an oracle $\mathcal{O}$ solving task $\mathcal{T}$ to solve another task $\mathcal{T}'$ is c-adaptive if it can be put in the following form. R consists of algorithms $R_1, \ldots, R_c, M$ such that:

1. Upon input z for the task $\mathcal{T}'$ and random coins ω, R_1 generates oracle queries $q_1^1, \ldots, q_k^1$ (for $k = \text{poly}(n)$) for task $\mathcal{T}$ using z, ω and obtains answers $a_1^1, \ldots, a_k^1$.

2. For $i \geq 2$, R_i generates queries $q_1^i, \ldots, q_k^i$ based on z, ω, and all the $a_1^j, \ldots, a_k^j$ for $j < i$ that it received from the oracle. It queries $q_1^i, \ldots, q_k^i$ to the oracle and obtains $a_1^i, \ldots, a_k^i$.

3. After obtaining all the queries and responses in round $1, \ldots, c$, M uses z, ω and a_k^j for $1 \leq j \leq c,\ 1 \leq i \leq k$, and solves the task $\mathcal{T}'$.

We also call a reduction *non-adaptive* if it is 1-adaptive.

In particular, the constructions and security analyses of cryptographic reductions can be adaptive. Most reductions in theoretical computer science are non-adaptive, and therefore understanding non-adaptive reductions goes a long way towards understanding how standard techniques might be used to solve a problem. There are however a few notable exceptions of important adaptive reductions. A (non-exhaustive) list of

such exceptions include the uniform security reduction of the construction that uses one-way functions to build pseudorandom generators [HILL89], lattice-based cryptography [Ajt96, MR04], and boosting [Sch89, Fre90, FS97].

Chapter 2

Computational learning through new lenses

2.1 Introduction

In terms of real-world impact as well as providing inspiration for new techniques and problems, two of the most successful areas of theoretical computer science are cryptography and computational learning theory. Cryptography, literally "secret writing" in Greek, originally addressed how to encode messages sent via an insecure channel so that no eavesdropper can learn anything about the message. Although long treated as an art, modern cryptography is a rigorous science and gives us a way to formally *prove* the security of cryptosystems. Its systematic study has allowed us to extend the cryptographic methodology of rigorous proofs of security in previously unimaginable ways, developing such notions as zero-knowledge proofs [GMR85]. Scientifically, techniques and ideas first developed in cryptography have successfully contributed back to other areas of theoretical computer science, especially in complexity theory.

Computational learning theory has a much more recent but nonetheless illustrious history. Although related to and drawing from predecessors in statistics such as Bayesian inference, it has been studied systematically only since the 1980's starting with a seminal paper of Valiant [Val84], where he proposed the Probabilistically Approximately Correct (PAC) model of learning. In this and most subsequent learning models, the learning algorithm is given examples labeled either "yes" or "no" according to some hidden labeling function (often called a *concept*), and the algorithm should use these examples to learn how to label future, unlabeled examples the same way as the hidden function would. Learning theory has seen many successes, such as the Boosting technique of Schapire and Freund [Sch89, Fre90, FS97], which allows us to convert algorithms that "weakly learn" into algorithms that learn almost perfectly. These techniques have been applied successfully in various areas such as natural language processing [CK05], medical diagnosis [MFLS01], and optical character recognition [DSS93], and boosting-based algorithms are deployed in many automated systems.

For a long time computer scientists have recognized that cryptography and learning are intuitively "inverses" of each other, as was discussed in a survey of Rivest [Riv93], which highlights connections known at the time. Roughly, sending messages secretly implies that no (computationally bounded) adversary can "learn" anything about the messages, and conversely if there are concepts that are hard to learn then perhaps one can use them as building blocks in a provably secure cryptosystem.

The main relationship between cryptography and learning explored in the literature was how hard cryptographic problems imply hard learning problems. For example, even in the paper defining the PAC model, Valiant observed that the existence of pseudorandom functions imply that polynomial-size circuits are hard to learn [Val84].

This connection has been refined over the years, as specific cryptographic assumptions have shown that specific (and often very simple) classes of concepts are hard to learn [KV89, PW90].

There were also efforts in the opposite direction, which used hard learning problems to build cryptographic primitives [BFKL93, IL90]. However, these works required that the model of PAC learning be modified so that hard instances of the learning problem be efficiently samplable. Unfortunately, no such connections are known when working in the standard model, where hard instances may exist but may not be efficiently samplable.

In the next few chapters (Chapter 2 through Chapter 5) we further investigate cryptography and computational learning theory by looking at their relationship through two new lenses: zero knowledge and black-box reductions. We will show that the hardness of learning is closely related to the existence of non-trivial zero knowledge proofs, which is a weaker form of cryptographic hardness than the existence of pseudorandom functions or one-way functions. We will also use the notion of black-box reductions, widely studied in complexity and especially in cryptography, to better understand how certain statements about learning may be proved. In Chapter 6, we will also look at an application of the inverse relationship: using learning algorithms to break cryptographic protocols in order to demonstrate that it is impossible to simultaneously achieve certain security and efficiency criteria.

2.1.1 Zero knowledge and learning

Zero knowledge is a notion of "learning nothing" that is modelled by the simulation paradigm: in an interactive protocol, a party learns nothing if it can produce a transcript of the protocol by itself that is indistinguishable from what it gets by

interacting with other parties. This fundamental idea was first proposed in the work of Goldwasser, Micali, and Rackoff [GMR85], and soon afterward it was proven by Goldreich, Micali, and Wigderson [GMW86] that assuming the existence of one-way functions, all languages in **NP** have a proof with this remarkable property.

More precisely, [GMW86] proved that for every language $L \in \mathbf{NP}$, one can use one-way functions to construct a protocol such that an all-powerful prover can convince a probabilistic polynomial-time verifier that an instance $x \in L$ without revealing *any* information other than the fact that $x \in L$. This zero knowledge guarantee holds because the verifier can efficiently simulate the entire interaction with the prover himself and produce a transcript of the interaction that is computationally indistinguishable from the transcript he would receive from an honest prover.

The class of languages with zero knowledge argument systems (which we call **ZK**) has since been studied in depth. Under the widely held conjecture that one-way functions exist, it is known that $\mathbf{ZK} = \mathbf{PSPACE}$ and hence since one-way functions imply $\mathbf{NP} \neq \mathbf{BPP}$, this in turn implies $\mathbf{ZK} \neq \mathbf{BPP}$ [BOGG+88]. Furthermore, in a long line of work [For87, AH91, OW93, Dam93, DGOW95, Oka96, SV97, GSV98, GV99, Vad04, OV07], we also know *unconditional* facts about **ZK**. For example, Ostrovsky and Wigderson [OW93] showed that if $\mathbf{ZK} \neq \mathbf{BPP}$, then there exist auxiliary-input one-way functions (AIOWF), which is a weak form of cryptographic hardness (see Definition 2.3.6). We will compare the complexity of PAC learning to the complexity of zero knowledge, and show that they are intimately connected.

2.1.2 Black box techniques

As many of the central questions in theoretical computer science seem out of reach of current techniques, one approach to better understand these questions is to bet-

ter understand whether various standard proof techniques might be able to resolve them. This approach was introduced from logic into computer science by Baker *et al.* [BGS75], where they proved that the **P** vs. **NP** question cannot be resolved via relativizing techniques. Since then, the technique has been taken to heart by computer scientists. Of special note to the cryptography community is the work of Impagliazzo and Rudich [IR89], who proved that relativizing techniques cannot possibly use the existence of one-way functions to construct key agreement.

In this work, we apply this philosophy to questions about the complexity of learning. We show that relativizing and black-box techniques are unable to resolve several central questions about learning. In particular, we show that one of the first questions raised by Valiant, namely whether learning in the PAC model is **NP**-hard, is unlikely to be proven by a large class of standard black-box techniques.

2.1.3 Overview of results and techniques

Our first main result is to prove that zero knowledge is "strictly" easier than PAC learning:

Theorem 2.1.1 (**ZK** is easier than PAC learning, informal)**.** *If* $\mathbf{ZK} \neq \mathbf{BPP}$*, then learning in the PAC model is hard. However, it is unlikely that standard techniques can use an oracle deciding any* $L \in \mathbf{ZK}$ *to successfully learn in the PAC model.*

This theorem is formalized in Theorem 3.2.1 and Theorem 4.2.1.

The first implication is proven by observing that Valiant's observation that pseudorandom functions (and equivalently one-way functions) imply learning is hard is actually "too strong" in the sense that the existence of pseudorandom functions implies that it is possible to *efficiently sample* hard-to-learn concepts. This is too strong

in some sense because learning is hard even if hard-to-learn concepts exist but are not efficiently samplable. By considering auxiliary-input one-way functions (AIOWF), we are still able to carry through Valiant's argument that learning is hard, although the hard-to-learn concepts may no longer be efficiently samplable. Using the Ostrovsky-Wigderson theorem that $\mathbf{ZK} \neq \mathbf{BPP}$ implies the existence of AIOWF, this gives the implication.

The second implication is proven by showing an oracle relative to which $\mathbf{ZK} = \mathbf{BPP}$ yet PAC learning is hard. Actually, the oracle proves the even stronger statement that AIOWF do not exist, yet PAC learning is hard. This oracle takes advantage of the fact that the inverter for an AIOWF knows the code that computes the AIOWF, while the learning algorithm in the PAC model does not know how the distribution of labeled examples are sampled (and indeed the distribution of labeled examples may not even be efficiently samplable). This rules out relativizing reductions. Next, we consider "GMW-style reductions", which build zero knowledge proofs for **NP** based on hardness of learning. In such a reduction, the prover, verifier, and simulator all have black-box access to a concept class, and the proof system should be zero knowledge provided that the concept class is hard to learn. We prove that such "GMW-style reductions" proving $\mathbf{NP} \subseteq \mathbf{ZK}$ cannot exist, unless $\mathbf{NP} \subseteq \mathbf{SZK}$ and the polynomial hierarchy collapses.

The second main result is to prove that PAC learning is "strictly" easier than **NP**

Theorem 2.1.2 (PAC learning is easier than **NP**, informal)**.** *If PAC learning is hard, then* $\mathbf{P} \neq \mathbf{NP}$*. However, it is unlikely that standard techniques can use an oracle that learns in the PAC model in order to decide an* **NP***-complete language.*

In fact, we will prove this theorem not only for PAC learning but for the stronger notion of *agnostic learning* [KSS92]. The first claim of the theorem is standard folklore,

while the second is formalized in Theorem 5.2.3 and Theorem 5.3.4.

The proof relies on two main observations. First, reductions that use a learning oracle to decide an **NP**-complete language must be efficient, and therefore the distributions of labeled examples that the reduction queries to its learning oracle must be efficiently samplable. Second, using an AIOWF inverter it is possible to learn any efficiently samplable distribution. This means that a reduction that uses a learning oracle to decide **NP** can be transformed into a reduction that uses an AIOWF inverter and decides **NP**. Such a reduction would lead to a surprising consequence, namely a proof that the average-case hardness of **NP** implies the existence of one-way functions, which collapses "Pessiland" and "Minicrypt" in Impagliazzo's taxonomy of worlds. If the reduction is further constrained to be non-adaptive and "strongly-black-box", such a reduction would in fact collapse the polynomial hierachy, and so such a reduction is considered implausible since the polynomial hierarchy is conjectured not to collapse.

Alternative notions of learning: in order to prove the above results, we explore notions of learning that are inpired by and related to the PAC model. We modify the PAC model in two ways: first consider the restricted case of learning where the distribution of labeled examples is *efficiently samplable*, *i.e.* there is an efficient circuit that samples from the distribution of labeled examples. We will give this sampling circuit to the learner as an additional input, thereby possibly making its task easier. We call this problem "circuit learning" or CircLearn for short. Next, we relax the model by changing the search problem to a decision problem. Instead of requiring the algorithm to find a hypothesis that labels similarly to the labeled examples, we only ask that the algorithm decide whether or not a good hypothesis *exists*. The relaxation along both these dimensions (requiring the distribution of labeled examples be efficiently samplable and requiring the learning algorithm only

to decide whether or not a good hypothesis exists) is called "circuit consistency" or CircCons for short.

The relationship between CircCons, CircLearn and AIOWF will underpin the proofs of our two main results above. In particular, we prove the following about these problems:

Theorem 2.1.3 (Informal)**.** CircCons $\in$ **ZK**.

This is formalized in Theorem 4.3.7. Notice this implies that if CircCons is hard, then **ZK** $\neq$ **BPP**. This stands in sharp contrast to Theorem 2.1.1, since we said that the hardness of the standard notion of PAC learning cannot imply **ZK** $\neq$ **BPP** via standard techniques.

Similarly,

Theorem 2.1.4 (Informal)**.** *If* CircLearn *is hard, then AIOWF exist.*

This is formalized in Corollary 3.5.2. Again this stands in contrast to the standard notion of PAC learning, where such an implication does not hold and cannot be proven via standard techniques.

Although CircCons, CircLearn are not "natural" learning problems since rarely in the real world does one know how the distribution of labeled examples is generated, nevertheless our results suggest that they are valuable notions to study as they shed light on the standard notion of learning.

2.1.4 Historical notes

The results of Chapter 2 through Chapter 5 are based on the works in [ABX08, Xia09]. [ABX08] focused on the relationship between learning and **NP**, while [Xia09] focused on the relationship between learning and **ZK**.

2.2 Definitions of computational learning

Let F be a collection of functions from $\{0,1\}^n \to \{0,1\}$, often called a concept class. An *example oracle* for (X,Y) where X,Y are distributions over $\{0,1\}^n$ and $\{0,1\}$ respectively is a randomized function that takes input 1^n and outputs a random sample (x,y) from the joint distribution X,Y. The notion of an example oracle models the intuitive notion of a "teacher": the teacher gives us labeled examples from which we should try to extract the underlying hidden labeling function. Of course, giving the learning algorithm access to an example oracle model is entirely equivalent to giving the learning algorithm an explicit (suitably large) set of labeled examples drawn independently from (X,Y).

First define the following notation to denote the error that a function f incurs labeling the joint distribution (X,Y):

Definition 2.2.1. For a joint distribution (X,Y) over $\{0,1\}^n \times \{0,1\}$, the error of $f : \{0,1\}^n \to \{0,1\}$ with respect to (X,Y) is

$$\mathsf{err}((X,Y),f) = \Pr_{X,Y}[f(X) \neq Y]$$

For a class of functions F, define

$$\mathsf{err}((X,Y),F) = \min_{f \in F} \mathsf{err}((X,Y),f)$$

Now we are prepared to define PAC learning:

Definition 2.2.2 (PAC Learning)**.** A procedure A ε-learns the concept class F if the following holds for every $f \in F$ and every distribution X over $\{0,1\}^n$. Given access to an example oracle for $(X, f(X))$, A produces with success probability $\geq 1 - 2^{-n}$ an ε-good hypothesis h (represented as a circuit), namely $\mathsf{err}((X,f(X)),h) \leq \varepsilon$. Equivalently, $\Pr_X[h(X) = f(X)] \geq 1 - \varepsilon$.

Notice that we have not required that h be related to F in any way; even if F is a simple class of functions (say linear functions), the hypothesis h could be much more complex. If we require in addition that $h \in F$ then this gives us the notion of *proper* PAC learning:

Definition 2.2.3 (Proper PAC Learning)**.** A procedure A ε-learns the concept class F *properly* if it ε-learns F and in addition it always outputs a hypothesis $h \in F$.

We say that learning F is hard (resp. non-uniformly hard) if there exists some $\varepsilon = 1/\text{poly}(n)$ such that no efficient algorithm (resp. non-uniform algorithm) A that ε-learns F in time $\text{poly}(n)$. We say that F is hard *almost everywhere* (resp. non-uniformly hard) to learn if there exists some $\varepsilon = 1/\text{poly}(n)$ such that every efficient algorithm fails to ε-learn F for all but finitely many n.

We say that PAC learning is hard (without specifying F) if learning $F = \mathsf{SIZE}(n^2)$ is hard. Define PAC learning to be almost always hard similarly. By a standard padding argument, learning $\mathsf{SIZE}(n^2)$ is hard (or hard almost everywhere) if and only if learning $\mathsf{SIZE}(n^c)$ for every constant $c > 0$.

Our motivation for defining PAC learning to be hard iff PAC learning $\mathsf{SIZE}(n^2)$ is hard is two-fold: first polynomial-size circuits are very powerful and so it is believable that they are hard to learn. Second, they are "unstructured" because they are universal, and therefore results about the hardness of learning $\mathsf{SIZE}(n^2)$ shed light about the *model* of PAC learning. If we had decided to study more structured concept classes (such as DNF or halfspaces) then the results we would have obtained would most likely have been based on the structure of those particular classes, not about the PAC learning model itself.

The model of learning can be relaxed to allow the learning algorithm to query the hidden labeling functions at points of its choosing. We say in such a case that the

learning algorithm has access to a *membership oracle* or is allowed to make *membership queries*.

Definition 2.2.4. We say that a PAC learning algorithm A gets access to a membership oracle if, when given access to an example oracle for $(X, f(X))$, the oracle is also allowed to query $f(x)$ on x of its own choosing.

Theorem 2.2.5 ([Val84, HILL89, GGM86])**.** *If one-way functions exist, then PAC learning is hard.*

This follows from the transformation of OWF's into PRF's via [HILL89, GGM86], and then it is clear from the definition of PRF's that $\{f_k\}$ form a concept class that is hard to learn.

Agnostic learning

Kearns *et al.* [KSS92] considered the *agnostic model*, where the example oracle (X, Y) may not correspond exactly to labeling by a function in F, namely there is no $f \in F$ such that $Y = f(X)$. In this setting, the best we can hope for is that a learner outputs a hypothesis that labels X almost as well as the best function in F.

Definition 2.2.6 (Agnostic learning)**.** A procedure A ε-agnostic learns a concept class F if given access to any example oracle (X, Y), A outputs with success probability $\geq 1 - 2^{-n}$ an ε-good hypothesis h, namely one satisfying $\mathsf{err}((X, Y), h) \leq \mathsf{err}((X, Y), F) + \varepsilon$.

Notice that in the particular case that (X, Y) is an example oracle of the form $(X, f(X))$ for some $f \in F$, agnostic learning is identical to PAC learning.

Efficiency and hardness are defined exactly as with PAC learning. As before, we say

that agnostic learning is hard (or hard almost everywhere) if learning $\mathsf{SIZE}(n^2)$ is hard.

The following proposition follows from the definitions.

Proposition 2.2.7. *If PAC learning is hard, then agnostic learning is hard.*

Notice that in our definitions of learning we require error to be $\leq \varepsilon$ for some $\varepsilon = 1/\text{poly}(n)$, while we ask that the success probability (*i.e.* the probability with which the algorithm A outputs a good hypothesis) to be $1 - 2^{-n}$. This is because the success probability can be amplified from any $1/\text{poly}(n)$ to $1 - 2^{-n}$ in polynomial time, simply by repeating the the learning algorithm many times independently to obtain several candidate hypotheses, then testing the accuracy of these candidates, and outputting the one that is most accurate. The analysis of this amplification follows by a straight-forward application of the Chernoff bound.

Proposition 2.2.8. *Suppose A outputs an ε-good hypothesis with probability $\delta > 0$ in either the PAC or agnostic model. Then for any polynomial $p(n)$ there exists an efficient A' using oracle calls to A that outputs an 2ε-good hypothesis with probability $1 - 2^{-p(n)}$ running in time $\text{poly}(p(n), 1/\delta)$.*

On the other hand, boosting the *accuracy* of the learning procedure is highly non-trivial. This technique is known as *boosting* [Sch89, Fre90, FS97] and can reduce the error from any $\frac{1}{2} - \gamma$ to any ε in time $\text{poly}(n, 1/\gamma, 1/\varepsilon)$. We will not require boosting to prove any of our results.

2.2.1 Oracles, black-boxes, and learning

We say that learning is hard relative to an oracle $\mathcal{O}$ if $\mathsf{SIZE}^{\mathcal{O}}(n^2)$ (or any subset thereof) is hard to learn for algorithms with oracle access to $\mathcal{O}$.

We say that an algorithm A uses black-box access to a concept class F if it satisfies the following. Let $|F| = 2^s$, and suppose each $f \in F$ maps $\{0,1\}^n \to \{0,1\}$. Let $\mathcal{O} : \{0,1\}^s \times \{0,1\}^n \to \{0,1\}$ be any oracle such that each $f \in F$ corresponds to some $\mathcal{O}(z,\cdot)$, namely for each $f \in F$ there exists unique $z \in \{0,1\}^s$ such that $f(x) = \mathcal{O}(z,x)$ for all $x \in \{0,1\}^n$. Then A accomplishes some task using only black-box access to F if A accomplishes that task given access to *any* such an oracle $\mathcal{O}$.

2.3 One-way functions

The most basic object studied in cryptoraphy is a one-way function (OWF), which is a function that is easy to compute but hard to invert.

Definition 2.3.1 (One-way functions)**.** A function $f : \{0,1\}^n \to \{0,1\}^m$ is a *one-way function* against uniform inverters (resp. non-uniform inverters) if f is efficiently computable by a uniform algorithm and for every efficient uniform (resp. non-uniform) algorithm I,

$$\Pr_{x \xleftarrow{R} U_n}[I(y) \in f^{-1}(y) \mid y = f(x)] < n^{-\omega(1)}$$

holds for all but finitely many input lengths n.

We will sometimes call this definition of OWF *standard* for emphasis and to distinguish from the other variants of OWF that we will encounter.

It is known that OWF are equivalent to many other notions of cryptography such as secret-key encryption and digital signatures [IL89, Rom90]. Here, we will use the following cryptographic primitives and the fact that their existence is equivalent to the existence of OWF.

Definition 2.3.2 (Pseudo-random functions (PRF))**.** A function $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ is a pseudorandom function against uniform (resp. non-uniform) distinguish-

ers if f is efficiently computable and for every efficient uniform (resp. non-uniform) oracle algorithm D, we have

$$\left| \Pr_{k \leftarrow_R U_n}[D^{f_k}(1^n) = 1] - \Pr_{\phi}[D^{\phi}(1^n) = 1] \right| \leq n^{-\omega(1)}$$

where $f_k = f(k, \cdot)$ and ϕ is a truly random function from $\{0,1\}^n \to \{0,1\}$.

In general PRF's can be defined with multi-bit outputs, but we will only encounter single-bit output PRF's.

Definition 2.3.3 (Distributional OWF (DOWF))**.** A function $f : \{0,1\}^n \to \{0,1\}^m$ is a distributional OWF against uniform (resp. non-uniform) inverters if f is efficiently computable and there exists a polynomial $p(n)$ such that for every efficient uniform (resp. non-uniform) algorithm I, it holds that

$$\Delta((x, f(x)), (I(y), y \mid y = f(x))) > 1/p(n)$$

over the random choice of $x \leftarrow_R U_n$ and the random coins of I.

It was proven by Håstad *et al.* [HILL89] and Goldreich *et al.* [GGM86] that PRF exist if and only if OWF exist.

Theorem 2.3.4 ([HILL89, GGM86])**.** *OWF exist if and only if PRF exist.*

Clearly, a distributional OWF is also a standard OWF, and Impagliazzo and Luby [IL89] showed that the converse holds as well.

Theorem 2.3.5 ([IL89])**.** *Distributional OWF exist if and only if (standard) OWF exist. Furthermore, the security analysis of the reduction that uses a OWF to build a distributional OWF is non-adaptive.*

2.3.1 Auxiliary-input one-way functions:

One-way functions are *uniformly* computable: there is a single Turing machine that computes f on all input lengths. This makes sense if we want to use them to build cryptographic systems since one wants to be able to compute f on all input lengths. From a complexity-theoretic point of view, however, one can relax this definition so that the function is *non-uniformly* computable: there exists a non-uniform family of circuits $\{C_n\}$ such that C_n computes f on inputs of length n. One can then relax this even further so that the particular family of circuits depends on the inverter it is trying to fool: this leads us to so-called auxiliary-input one-way functions, which were introduced in the work of Ostrovsky and Wigderson [OW93].

Definition 2.3.6. Auxiliary-input one-way functions (AIOWF) against uniform (resp. non-uniform) inverters exist if for every uniform (resp. non-uniform) inverter I, there exists an infinite collection W of functions such that for every function $f \in W$ mapping $\{0,1\}^n \to \{0,1\}^m$, f is computable by a circuit of size $s(n) = \text{poly}(n)$, and it holds that

$$\Pr_{x \leftarrow_R U_n}[I(f,y) \in f^{-1}(y) \mid y = f(x)] < s^{-\omega(1)}$$

Whereas with standard one-way functions the hard-to-invert function is fixed, here the family of hard functions W may depend on the inverter I. Because of this, notice above that the inverter also takes a description of the function (as a circuit) as an auxiliary input.

Similarly, we can define auxiliary-input variants of the PRF and DOWF.

Definition 2.3.7. Auxiliary-input pseudorandom-functions (AIPRF) against uniform (resp. non-uniform) distinguishers exist if for every uniform (resp. non-uniform) oracle distinguisher D, there exists an infinite collection W of functions where for ev-

ery $f \in W$, $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$, f is computable by a circuit of size $s(n) = \text{poly}(n)$ and it holds that

$$\left| \Pr_{k \xleftarrow{R} U_n} [D^{f_k}(f, 1^s) = 1] - \Pr_{\phi}[D^{\phi}(f, 1^s) = 1] \right| \leq s^{-\omega(1)}$$

where $f_k = f(k, \cdot)$ and ϕ is a truly random function from $\{0,1\}^n \to \{0,1\}$.

Definition 2.3.8. Auxiliary-input distributional OWF (AIDOWF) against uniform (resp. non-uniform) inverters exist if for every uniform (resp. non-uniform) inverter I, there exists a polynomial $p(s)$ and an infinite collection W of functions where for every $f \in W$, $f : \{0,1\}^n \to \{0,1\}^m$, f is computable by a circuit of size $s(n) = \text{poly}(n)$ and it holds that

$$\Delta((x, f(x)), (I(f, y), y \mid y = f(x))) > 1/p(s)$$

over random choie of$x \leftarrow_{\text{R}} U_n$ and the random coins of I.

Remark 2.3.9. Theorem 2.3.4 and Theorem 2.3.5 both extend in a straight-forward way to show that AIOWF exist if and only if AIPRF exist if and only if AIDOWF exist.

Furthermore, these equivalences are constructive. To illustrate what we mean by constructive in the particular the case of building AIOWF from AIDOWF, we mean that there exists a pair of efficient oracle algorithms $R^{(\cdot)}_{\mathsf{cons}}, R^{(\cdot)}_{\mathsf{sec}}$ (the construction and security reduction, respectively; see Section 1.3 for background on constructions and security reductions) such that the following holds. For every algorithm A (which is supposed to invert the AIOWF), let R^A_{sec} be the security reduction applied to A (R^A_{sec} is supposed to invert AIDOWF). If AIDOWF exist, then there exists a set of functions W that is distributionally hard to invert for R^A_{sec}. Let $R_{\mathsf{cons}}(W) = \{R^C_{\mathsf{cons}} \mid C \in W\}$, which denotes set of circuits obtained by applying the construction applied to each

circuit in W. Then $R_{\mathsf{Cons}}(W)$ is a family of functions that is hard to invert (in the sense of AIOWF) for A.

2.4 Zero knowledge

2.4.1 Definitions

Let $\langle P, V\rangle(x)$ denote the verifier's view of the transcript of an interactive protocol between a prover P and a verifier V on common input x. This consists of all messages sent by the prover and verifier, as well as the verifier's random coins.

Definition 2.4.1. We say that $L \in \mathbf{ZK}$ if there exists an efficient (randomized) verifier strategy such that the following hold:

- Completeness: $\forall x \in L$, there is a prover strategy such that V accepts the transcript $\langle P, V\rangle(x)$ with probability $1 - 2^{-n}$.
- Soundness: $\forall x \notin L$, for any efficient prover strategy P^*, V accepts the transcript $\langle P^*, V\rangle(x)$ with probability at most 2^{-n}.
- Zero knowledge: there exists an efficient simulator S such that $\forall x \in L$, the distribution $\langle P, V\rangle(x)$ is computationally indistinguishable from $S(x)$.

Notice that we only define zero knowledge with respect to an *honest* verifier strategy, *i.e.* the verifier does not deviate from the protocol. It is known [GSV98, Vad04, OV07] that honest-verifier protocols can be compiled into protocols that are also zero knowledge with respect to cheating verifiers. For this paper we will simply work with the honest-verifier versions of zero knowledge.

The definition of zero knowledge can be tailored in several ways. We say that a protocol satisfies statistical zero knowledge if the zero knowledge property holds against unbounded distinguishers, namely

$$\Delta(S(x), \langle P, V \rangle(x)) \leq \varepsilon(n)$$

where ε is a negligible function. We say that a protocol is a proof if soundness must hold against unbounded prover strategies. This leads to the four following variants of zero knowledge:

1. **SZKP**: statistical zero knowledge proofs
2. **CZKP**: computational zero knowledge proofs
3. **SZKA**: statistical zero knowledge arguments
4. **CZKA**: computational zero knowledge arguments

By definition it holds that **SZKP** $\subseteq$ **CZKP**, **SZKP** $\subseteq$ **SZKA**, **CZKP** $\subseteq$ **CZKA**, and **SZKA** $\subseteq$ **CZKA**.

Remark 2.4.2. Unless otherwise specified, throughout this thesis we let **SZK** denote **SZKP** and **ZK** denote **CZKA**.

2.4.2 Previous results

Zero knowledge has been deeply studied in the literature, and we will take advantage of these previous results to aid in our own study of zero knowledge as it relates to learning. The following theorems will be important tools used in this paper.

Theorem 2.4.3 ([For87, AH91])**.** **SZK** $\subseteq$ **AM** $\cap$ **coAM**

Theorem 2.4.4 ([Ost91, OW93]). **ZK** $\neq$ **BPP** *implies that there exist AIOWF against uniform algorithms.*

Definition 2.4.5 ([SV97]). Statistical Difference with parameters α, β (denoted $\mathsf{SD}_{\alpha,\beta}$) is the following promise problem: A YES instance is a pair of circuits sampling two independent distributions X, Y such that $\Delta(X,Y) > \alpha$. A NO instance is a pair of circuits sampling two independent distributions X, Y such that $\Delta(X,Y) < \beta$.

Theorem 2.4.6 ([SV97, Vad04]). $\mathsf{SD}_{\alpha,\beta}$ *for any constants* $\alpha^2 > \beta$ *is* **SZK***-complete.*

Theorem 2.4.7 ([Vad04, OV07]). *A promise problem* $\Pi = (\Pi_Y, \Pi_N) \in$ **ZK** *if and only if there exists an efficient reduction* Red*, a set* $W \subseteq \Pi_Y \cup \Pi_N$*, and an efficiently computable function* f *mapping* W *to circuits such that the following hold.*

1. *The reduction* Red *reduces the promise problem* $\Pi' = (\Pi_Y \setminus W, \Pi_N \setminus W)$ *to* $\mathsf{SD}_{\alpha,\beta}$ *(for some constants* $\alpha^2 > \beta$*).*
2. *Let* f_z *denote the function computed by the circuit* $f(z)$ *for* $z \in W$*. Then the family of functions* $\{f_z \mid z \in W\}$ *is hard to invert for all non-uniform inverters.*

Furthermore, if $W = \varnothing$ *then* $\Pi \in$ **SZKP***, if* $W \subseteq \Pi_Y$ *then* $\Pi \in$ **CZKP***, and if* $W \subseteq \Pi_N$ *then* $\Pi \in$ **SZKA**.

2.4.3 Relativizing definitions of zero knowledge

Since we will study black-box constructions of zero-knowledge protocols, we will work with relativized versions of **ZK**. We say $L \in \mathbf{ZK}^{\mathcal{O}}$ if it satisfies the definition of **ZK** as defined above except the prover, verifier, simulator, and distinguisher are all allowed access to the oracle $\mathcal{O}$. Also, $\mathsf{SD}^{\mathcal{O}}_{\alpha,\beta}$ is like $\mathsf{SD}_{\alpha,\beta}$ except circuits are allowed $\mathcal{O}$ gates.

Remark 2.4.8. Examining the proofs of the above Theorem 2.4.4 and Theorem 2.4.6, we observe that they all relativize.

Furthermore, one direction of Theorem 2.4.7 is relativizing: for any oracle $\mathcal{O}$, if a problem $\Pi \in \mathbf{ZK}^{\mathcal{O}}$, then there exist a reduction Red, W, and f as stated in the theorem, where Red reduces $\Pi' = \Pi \setminus W$ to $\mathsf{SD}^{\mathcal{O}}_{\alpha,\beta}$ and f maps W to a family of circuits containing $\mathcal{O}$ gates. Letting $f^{\mathcal{O}}_z$ denote the function computed by the circuit $f(z)$ with oracle gates, the family of functions $\{f^{\mathcal{O}}_z \mid z \in W\}$ is hard to invert against all non-uniform inverters, even those containing $\mathcal{O}$ gates.

On the other hand, the other direction of Theorem 2.4.7, which shows that any language satisfying the characterization is in **CZKP**, uses non-relativizing techniques. This is because this direction uses the non-relativizing proof that 3-COL is in **ZK** assuming some form of commitment scheme [GMW86].

2.5 Usage of diagrams

We will often employ diagrams to show the relationship between different forms of hardness. See Figure 2.5 for an example. Each box represents some form of computational hardness, and an arrow from box A to box B with a circle means that A implies B; an open circle indicates that the reduction holds, and a crossed out circle indicates that such reductions do not exist or their existence would imply consequences that are surprising or contradict standard conjectures. The reduction types are:

1. "FB" indicates fully-black-box reductions.
2. "CB" indicates construction-black-box reductions.
3. "Rel" indicates relativizing reductions.

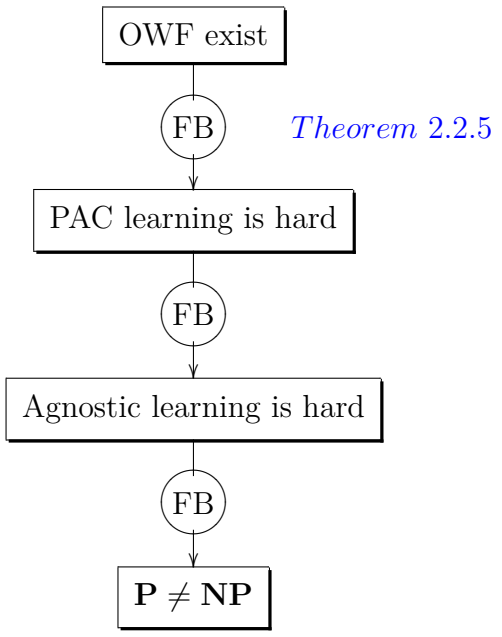

Figure 2.1: Previously known relationships

4. "∀∃" indicates ∀∃-construction-black-box reductions.

5. "A" implies arbitrary reductions (not falling into any of the above categories).

6. An asterisk "∗" means that further restrictions apply, and the reader is asked to refer to the listed theorem for the precise statement.

The arrows are annotated with the theorem that proves that implication; arrows lacking annotations either follow immediately from the definitions, or are considered folklore.

For example, Figure 2.5 says that if one-way functions exist then learning is hard, this implication can be proven using a black-box reduction, and is stated in Theorem 2.2.5.

Chapter 3

Learning and one-way functions

One of the earliest connections between learning and other areas of computer science was Theorem 2.2.5, which states that the existence of one-way functions, using the transformation of one-way functions into pseudo-random functions [HILL89, GGM86], imply that learning polynomial-size circuits is hard. Further work by Kearns and Valiant [KV89] showed that if certain concrete cryptographic assumptions such as the RSA assumption or the hardness of factoring hold, then learning even very weak classes such as Boolean formulae and constant-depth threshold circuits is hard.

In this chapter we continue the inquiry into connections between learning and one-way functions, and show that learning is related not just to one-way functions but also to auxiliary-input one-way functions. In order to obtain a finer understanding we introduce several notions of learning related to the PAC model, and we show how these notions of learning relate to AIOWF.

The chapter is organized as follows. In Section 3.1 we define a decisional version of the PAC learning problem, and in Section 3.2 we observe that if AIOWF exist then this decisional version of PAC learning is hard. In Section 3.3 we show that the

converse implication cannot be proven by standard techniques by exhibiting an oracle such that learning is hard but AIOWF do not exist.

Then in Section 3.4 we define some new problems (CircCons and CircLearn) that are related to the PAC learning problem. Essentially these problems require that the distribution of labeled examples that the learning algorithm sees be efficiently samplable, and furthermore the learning algorithm gets the circuit sampling the the distribution as an additional input. In Section 3.5 we show that if CircLearn is hard, then AIOWF exist. This stands in contrast to the results of Section 3.3, which rules out such implications for the standard definitions using standard techniques.

The new notions CircCons and CircLearn introduced in this chapter along with the theorems relating them to AIOWF will serve as tools to prove results relating (standard) PAC learning to **ZK** and **NP** in subsequent chapters.

A more detailed summary outlining the results proven in this chapter is included at the end of the chapter.

3.1 A decisional version of learning

To better understand the PAC model, we develop several related notions of learning. In defining our new notions, there are two main features of the PAC model that we will modify: the *search nature* of the model and the *oracle nature* of the model. Here we first consider the search nature of the problem, and we consider the oracle nature of the problem in Section 3.4.

PAC learning is inherently a *search* problem: given an example oracle that generates examples labeled according to some hidden function, find a hypothesis that labels almost all examples the same way as the hidden function does. Specifically, the

learning algorithm must produce such a hypothesis and not simply claim that it exists.

It is well-known that many search problems reduce to their decisional version. The most famous example is SAT: given an algorithm that can decides whether or not a given Boolean formula is satisfiable, one can also efficiently *find* a satisfying assignment of that formula. This phenomenon is called downward self-reducibility and appears throughout computational complexity, but it is by no means shared by all computational problems, *e.g.* deciding primality is easy, but integer factorization is believed to be hard.

One can therefore ask about a *decisional* version of learning: given an example oracle, does there *exist* a function in some target concept class F that labels examples the same way as examples generated by the oracle? In this section we formalize this decisional model, and we will see that Goldreich *et al.* showed that under certain assumptions, the decision problem may be easier than the search problem (Theorem 3.1.4).

We begin with the following definition, which is called "general property testing" by Goldreich *et al.* [GGR98].

Definition 3.1.1 (Testing proper PAC β-consistency)**.** *A tests for proper PAC β-consistency of a concept class F if given access to any distribution of labeled examples (X, Y) the following holds.*

- YES instance: if there exists $f \in F$ such that $Y = f(X)$, then A outputs 1 with probability $1 - 2^{-n}$.
- NO instance: if $\mathsf{err}((X, Y), F) > \beta$, then A outputs 0 with probability $1 - 2^{-n}$.

Notice this is a promise problem: there are example oracles which satisfy neither the

YES nor NO conditions, in which case we do not care what A outputs. We first explore the intuition behind this definition and why we call it *proper*. We want the definition to classify example oracles into those for which the task of learning (with respect to the target concept class F) is possible and those for which the task is impossible.

Looking at Definition 3.1.1 again, the definition of YES instances is the obvious one. The definition of NO instances is also what we would expect, namely the labeling given by (X, Y) to be very different from any labeling in F. Why then do we emphasize that this corresponds to the notion of proper learning?

One property we want from the definition is for it to be related to the standard PAC learning definition in the following way: if there exists a PAC learning algorithm for learning F in the standard sense, then the following reduction PACtoTesting should be a good tester for the PAC consistency of F: to test the consistency of (X, Y), run the PAC learning algorithm on (X, Y) to obtain a hypothesis h. Then sample more examples (x, y) from the example oracle and check if $h(x) = y$; if a $1 - \beta/2$ fraction of these examples are labeled correctly output 1, otherwise output 0.

If we had an algorithm A that learned F properly, then clearly the above reduction would also give an algorithm for testing proper PAC consistency. However, suppose now that A learns F, but is not necessarily proper. We argue that this does not necessarily give us an algorithm to test proper PAC consistency: in particular suppose we are given an example oracle $(X, g(X))$ where g is very far from F, *i.e.* $\mathsf{err}((X, g(X)), F) > 1/2$, but g is still computable by a circuit of size n^3. Then it is possible that A will output a circuit computing g even though g is far from F, in which case using PACtoTesting would give us the wrong answer. Therefore we propose the following definition of testing (not necessarily proper) PAC consistency:

Definition 3.1.2 (Testing PAC β-consistency)**.** *A tests for the PAC β-consistency of a concept class F if given access to any distribution of labeled examples (X, Y) the following holds.*

- YES instance: if there exists $f \in F$ such that $Y = f(X)$, then A outputs 1 with probability $1 - 2^{-n}$.
- NO instance: if $\mathsf{err}((X, Y), \mathsf{SIZE}(n^{\log\log n})) > \beta$, then with probability $1 - 2^{-n}$ A outputs 0.

The term $n^{\log\log n}$ can be replaced by any super-polynomial function without affecting any of our results. The definition of testing consistency can also be augmented to give the tester access to a membership oracle (*i.e.* A can query $f(x)$ for x of its choosing), or restricted to so that the tester only needs to succeed over specific classes of distributions X (*e.g.* $X = U_n$ the uniform distribution).

Testing PAC consistency of F is hard against uniform (resp. non-uniform) algorithms if there exists some $\beta = 1/\mathrm{poly}(n)$ such that no uniform (resp. non-uniform) algorithm can test PAC β-consistency of F in time $\mathrm{poly}(n)$, and we say that testing PAC consistency is hard if testing PAC consistency of $\mathsf{SIZE}(n^2)$ is hard.

Testing PAC 1/2-consistency is clearly trivial (always output 1, since every (X, Y) fails the NO condition) but the problem gets harder as β gets smaller. The following proposition follows immediately from the definitions.

Proposition 3.1.3. *If testing PAC consistency is hard, then PAC learning is hard.*

It was shown by [GGR98] that under certain cryptographic assumptions, there exists concept classes for which it is easy to test PAC consistency but hard to PAC learn.

Theorem 3.1.4 ([GGR98]). *Assuming the existence of weak trapdoor one-way permutations with dense domains, there exists a concept class F such that testing PAC consistency for F is easy but PAC learning F is hard.*

Relationship to property testing

We named the decisional model "testing consistency" expressly to highlight the connection between it and the notion of *property testing*, which has been widely studied in the computer science literature [BLR90, RS96, GGR98]. In the property testing framework, one often wishes to test whether a Boolean function f satisfies a property P, where P is just a subset of all functions (for example, P can be the set of linear functions). One wants to distinguish between YES instances, which are functions $f \in P$, and NO instances, which are functions f that are ε-far from P, *i.e.* $\mathsf{err}((U_n, f(U_n)), P) \geq \varepsilon$. One has only black-box access to f and wants to minimize the number of queries made.

As observed by [GGR98], this is remarkably similar to the problem of testing β-consistency. In fact, testing for a property P is completely equivalent to testing proper PAC consistency of P where the input is restricted to distributions of the form $(U_n, \phi(U_n))$ for some function ϕ. [GGR98] explores more relationships between testing proper PAC consistency and PAC learning.

In testing (not necessarily proper) PAC consistency, the role of the property P is identical to the role of the concept class F, and the YES and NO conditions are very similar. There are however several important differences:

1. In testing PAC consistency, the NO instance is much more severe: the function must be far from any *efficiently computable function* and not just far from any function in F.

2. In the property testing setting an instance is just a function f and the NO condition applies to functions that are far from P with respect to inputs drawn from the uniform distribution. In testing PAC consistency an instance is an oracle (X, Y) and therefore the labeling Y may not even be a deterministic function of X, and furthermore the NO condition applies to labelings that are far from efficiently computable functions with respect to inputs drawn from the distribution X, not necessarily with respect to the uniform distribution.

3.2 AIOWF implies testing PAC learnability is hard

Theorem 2.2.5, which states that the existence of one-way functions implies learning is hard, gives "more than is necessary": not only is learning hard, but one can efficiently sample a function that is hard to learn: sample a key k for the pseudo-random function and let f_k be the function to be learned. This speaks to the difference between worst-case and average-case hardness: the existence of one-way functions is an average-case notion of hardness because it requires that one can efficiently sample hard instances of a hard problem. Learning on the other hand can (and we believe *should*) be viewed as a worst-case notion of hardness because it requires that there *exist* hard functions computable by polynomial-size circuits that are hard to learn, but it does not require those hard instances to be efficiently samplable. In fact, which instances are hard may even depend on which learning algorithm is being considered.

Already Blum *et al.* explicitly pointed out this discrepancy when trying to build cryptographic primitives from hard learning problems [BFKL93]. Malkin asked whether one can base the hardness of learning on weaker notions that better reflect the worst-case nature of hardness of learning [Mal08], and in this section we observe that AIOWF constitute one such notion. The proof that the existence of AIOWF im-

ply that learning is hard is essentially the same as the proof of Theorem 2.2.5, one simply needs to observe as in Remark 2.3.9 that AIPRF can be built from AIOWF. We reproduce the proof that AIPRF implies testing PAC consistency is hard in full detail for the sake of completeness.

Theorem 3.2.1. *If there exist AIOWF's against uniform (resp. non-uniform) distinguishers, then testing PAC* $(\frac{1-\varepsilon}{2})$*-consistency is hard against uniform (resp. non-uniform) algorithms, where* $\varepsilon = 2^{-n/4}$.

Proof. We say that the AIPRF are computable in size n^2 if for every distinguisher D, there exists an infinite collection W of functions such that each $f \in W$ is computable in size n^2 and D cannot distinguish f_k from a truly random function (as in Definition 2.3.7). A standard padding argument shows that the existence of AIPRF implies that there exist AIPRF computable in size n^2.

Suppose for the sake of contradiction thst testing PAC consistency were not hard. Then there exists an efficient A such that for every circuit f computable in size n^2, given access to the example oracle $(U_n, f(U_n))$, A outputs 1 with probability $1 - 2^{-n}$, and given oracle access to $(U_n, g(U_n))$ where $\mathsf{err}((U_n, g(U_n)), \mathsf{SIZE}(n^{\log\log n})) > \frac{1-\varepsilon}{2}$ it outputs 0 with probability $1 - 2^{-n}$.

We prove this contradicts the existence of AIPRF. Use A to construct B that, given access to an oracle $\mathcal{O}$, runs A on the example distribution $(U_n, \mathcal{O}(U_n))$ and outputs what A outputs. By the hypothesis that AIPRF exist, there exists a collection W of functions $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ each computable in size n^2 such that

$$\left| \Pr_{B, k \leftarrow_R U_n}[B^{f_k}(f, 1^n) = 1] - \Pr_{B,\phi}[B^{\phi}(f, 1^n) = 1] \right| \leq n^{-\omega(1)} \tag{3.2.1}$$

where ϕ is a truly random function. Pad f so that it is computable by a size n^2 circuit; this implies that for every k, f_k is computable by a n^2 size circuit as well.

By the construction of B, for any oracle $\mathcal{O}$, $B^{\mathcal{O}}(f, 1^n) = 1$ if and only if $A^{(U_n, \mathcal{O}(U_n))}(1^n) = 1$. Because f_k is computable in size n^2 for every k, it holds that $\Pr_k[B^{f_k}(1^n) = 1] \geq 1 - 2^{-n}$ by our assumption that A tests PAC consistency with size n^2 circuits.

On the other hand, since ϕ is a random function, with overwhelming probability ϕ is far from any function in $\mathsf{SIZE}(s)$ where we let $s = n^{\log \log n}$. More precisely, because the probability that a random function ϕ agrees with f is bounded by $\Pr_\phi[\mathsf{err}((U_n, \phi(U_n)), f) \leq \frac{1-\varepsilon}{2}] \leq 2^{-\varepsilon^2 2^n/8}$ because of a Chernoff bound, we can write:

$$\begin{aligned}
&\Pr_\phi[\exists f \in \mathsf{SIZE}(s),\ \mathsf{err}((U_n, \phi(U_n)), f) \leq \tfrac{1-\varepsilon}{2}] \\
&\quad \leq 2^{O(s \log s)} \Pr_\phi[\mathsf{err}((U_n, \phi(U_n)), f) \leq \tfrac{1-\varepsilon}{2}] \\
&\quad \leq 2^{O(s \log s) - \varepsilon^2 2^n/8} && \textit{(By Chernoff bound)} \\
&\quad \ll 2^{-n} && \textit{(Using } \varepsilon = 2^{-n/4}\textit{)}
\end{aligned}$$

For such ϕ, we have that $B^\phi(1^n) = 1$ with probability at most 2^{-n} by our assumption that A tests PAC consistency. Therefore, $\Pr_\phi[B^\phi(1^n) = 1] \leq 2^{-n+1}$. Putting these together, it holds that

$$\Pr_{B,k}[B^{f_k}(1^n) = 1] - \Pr_{B,\phi}[B^\phi(1^n) = 1] \geq 1 - 3 \cdot 2^{-n}$$

which contradicts Inequality 3.2.1. ■

3.3 An oracle separating learning and AIOWF

Given Theorem 3.2.1, it is tempting to think that one might be able to show that the existence of AIOWF is equivalent to the hardness of testing PAC consistency. In this section we show that this is probably not the case, and that relativizing techniques cannot use the hardness of testing PAC consistency to construct AIOWF. This immediately implies a separation of hardness of PAC learning and AIOWF.

Theorem 3.3.1. *There exists an oracle $\mathcal{O}$ for which testing PAC consistency with a membership oracle is hard, but AIOWF do not exist.*

Intuition behind Theorem 3.3.1: the intuitive difference between PAC learning and inverting AIOWF we exploit is that in PAC learning, the learner knows nothing about how the labeled examples (X, Y) are produced, whereas with AIOWF, the inverting algorithm *does* know a description of the function f it is trying to invert.

Our oracle will be defined using a distribution over functions $\mathcal{R}^{(n)} : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$. This defines the collection of functions $\{\mathcal{R}_z\}$ where for $z \in \{0,1\}^n$, we define $\mathcal{R}_z = \mathcal{R}^{(n)}(z, \cdot)$. For each $z \in \{0,1\}^n$, with probability $2^{-n/2}$ the distribution sets z to be a "hard instance", *i.e.* it sets $\mathcal{R}_z$ to be a uniformly random function, and with probability $1 - 2^{-n/2}$ it sets $\mathcal{R}_z$ to be the all zero function $\mathcal{R}_z \equiv 0$.

We show (in Lemma 3.3.5) that almost surely over the choice of $\mathcal{R}$, the concept class $F = \{\mathcal{R}_z\}_{z \in \{0,1\}^*}$ is hard to learn for non-uniform algorithms with $\mathcal{R}$ gates. The intuition is that there are roughly $2^{n/2}$ hard instances z on inputs of length n, and they are chosen at random, so no polynomial-size circuit can find all of them, and no circuit can learn hard instances it cannot find because hard instances are random functions. Notice that we must choose *many* hard instances because a circuit's non-uniformity can be specified *after* the oracle is chosen, and so the advice may reveal where some of the hard instances are hidden; by choosing $2^{n/2}$ hard instances, no polynomial amount of advice can specify all of the hard instances, and so for any polynomial-size circuit some hard instances remain random-looking.

The second condition is to check that AIOWF do not exist. One idea to assure this is to define another oracle $\mathcal{I}$ that inverts circuits with $\mathcal{R}$ gates. It is straight-forward to show that no non-uniform circuit family can learn F even given access to $\mathcal{I}$, but since $\mathcal{I}$ can invert all circuits with $\mathcal{R}$ gates, AIOWF do not exist. This type of

proof technique is common in the cryptographic literature (*e.g.* [HHRS07]) and rules out fully black-box reductions building AIOWF from hardness of learning. However, it does not rule out relativizing reductions, which allow the circuit *computing* the AIOWF to also use $\mathcal{I}$ gates: it is not at all obvious how or even if $\mathcal{I}$ can invert circuits that contain $\mathcal{I}$ gates. This distinction is not merely cosmetic: in particular, the Ostrovsky-Wigderson theorem (Theorem 2.4.4) is *not* fully black-box but it is relativizing (see Section 4.2.1 for a discussion of this distinction). Later we will combine Theorem 2.4.4 with Theorem 3.3.1 to conclude that there is an oracle that separates hardness of learning and $\mathbf{ZK} \neq \mathbf{BPP}$ (see Theorem 4.2.1). For this purpose, it is essential that we rule out relativizing reductions and not just fully black-box reductions. This requires a more powerful oracle, which we describe now.

Definition 3.3.2. A language L is in $\mathbf{PSPACE}^{\mathcal{R}}_*$ if there exists a pair (M_1, M_2) where M_1 is a polynomial-time Turing machine and M_2 is a polynomial-space oracle Turing machine such that $x \in L$ if and only if $M_1(x)$ outputs $z_1, \ldots, z_m \in \{0,1\}^*$ and $M_2(x)$ using only oracle gates $\mathcal{R}_{z_1}, \ldots, \mathcal{R}_{z_m}$ outputs 1.

There is a natural complete language $\mathsf{QBF}^{\mathcal{R}}_*$ for this class. $\mathsf{QBF}^{\mathcal{R}}_*$ is the language of satisfiable QBF where the final propositional formula is allowed $\mathcal{R}_z = \mathcal{R}^{(n)}(z, \cdot)$ gates, but only for *fixed* z (for example, "$\exists z, \mathcal{R}_z(x)$" is not a valid formula for $\mathsf{QBF}^{\mathcal{R}}_*$). It follows immediately from the proof that QBF is complete for $\mathbf{PSPACE}$ that $\mathsf{QBF}^{\mathcal{R}}_*$ is complete for $\mathbf{PSPACE}^{\mathcal{R}}_*$ (see Proposition A.1.1 for a proof).

Our separating oracle will decide $\mathsf{QBF}^{\mathcal{R}}_*$.

Definition 3.3.3. $\mathcal{O}$ is drawn from the following the distribution. First, for each n select a function $\mathcal{R}^{(n)} : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ by letting each $z \in \{0,1\}^n$ be a *hard instance* with probability $2^{-n/2}$, where we set $\mathcal{R}_z = \mathcal{R}^{(n)}(z, \cdot)$ to be a random function, and letting z be an *easy instance* with probability $1 - 2^{-n/2}$, where $\mathcal{R}_z \equiv 0$.

Let $\mathcal{O}$ decide $\mathsf{QBF}_*^{\mathcal{R}}$, which is $\mathbf{PSPACE}_*^{\mathcal{R}}$-complete.

Learning is still hard relative to $\mathcal{O}$: even with access to $\mathcal{O}$, the learner can only "see" $\mathcal{R}_z$ for polynomially many z because a $\mathsf{QBF}_*^{\mathcal{R}}$ formula can only contain $\mathcal{R}_z$ gates for fixed z, and so can contain $\mathcal{R}_z$ gates for at most polynomially many z. Since $\mathcal{O}$ does not help the learner find additional hard instances, the hard instances $\mathcal{R}_z$ that remain hidden also remain random-looking, and therefore hard to learn.

On the other hand, we can use $\mathcal{O}$ to build an inverter that inverts any AIOWF. Given any f as a circuit with $\mathcal{O}$ gates, we show that it is possible to use $\mathcal{O}$ to find "heavy queries", *i.e.* z such that the computation of $f(x)$ queries $\mathcal{R}_z$ with probability $\geq 1/\mathrm{poly}(n)$ over the choice of random x. Notice this means there can be at most $\mathrm{poly}(n)$ many heavy z. We show that if f only ever queried $\mathcal{O}$ on either easy or heavy z, then one can efficiently invert f using oracle queries only for the $\mathrm{poly}(n)$ heavy instances. Of course f may actually query $\mathcal{O}$ on "bad z" that are hard and yet not heavy, but we show that on a *typical* $y = f(x)$ where x is chosen at random, the computation of $f(x)$ is unlikely to call $\mathcal{O}$ on any bad z. Our inverter finds the heavy queries and then inverts pretending that f only calls $\mathcal{O}$ on good z, and we prove that this inverter succeeds with noticeable probability over random $y = f(x)$.

Theorem 3.3.1 follows immediately from the following even stronger statement.

Theorem 3.3.4. *With probability* 1 *over the choice of oracle* $\mathcal{O}$ *as in Definition 3.3.3, testing PAC* $(\frac{1-\varepsilon}{2})$*-consistency for* $F = \{\mathcal{R}_z\}_{z\in\{0,1\}^*}$ *over the uniform distribution and with membership queries is hard for any* $\varepsilon \geq 1/2^{\log^2 n}$*, but no AIOWF against uniform inverters exists.*

Proof. The theorem immediately follows from the following two lemmas, which we prove in the next two subsections.

Lemma 3.3.5. *With probability 1 over the choice of $\mathcal{O}$, testing PAC $(\frac{1-\varepsilon}{2})$-consistency for the concept class $F = \{\mathcal{R}_z\}_{z\in\{0,1\}^*}$ over the uniform distribution with membership queries is hard even if the tester has access to $\mathcal{O}$.*

Lemma 3.3.6. *There is an efficient oracle algorithm $I^{(\cdot)}$ that, with probability 1 over choice of $\mathcal{O}$ as in Definition 3.3.3, given any function $f : \{0,1\}^n \to \{0,1\}^m$ described as a circuit of size s with $\mathcal{O}$ gates, satisfies:*

$$\Pr_{x\leftarrow_R\{0,1\}^n}[I^{\mathcal{O}}(f^{\mathcal{O}}, y) \in (f^{\mathcal{O}})^{-1}(y) \mid f^{\mathcal{O}}(x) = y] > 1/2$$

■

Since testing PAC consistency is harder than PAC learning (Proposition 3.1.3), it follows that:

Corollary 3.3.7. *With probability 1 over the choice of oracle $\mathcal{O}$, PAC learning $F = \{\mathcal{R}_z\}_{z\in\{0,1\}^*}$ over the uniform distribution and with membership queries is hard, but no AIOWF against uniform inverters exists.*

In fact, this argument already rules out a more general class of proofs, namely $\forall\exists$-construction-black-box reductions. We defer this discussion to Section 3.3.3.

3.3.1 Testing consistency is hard relative to $\mathcal{O}$

Proof of Lemma 3.3.5. To prove this lemma, we show that any oracle circuit $C^{\mathcal{O}}$ has probability $2^{-2^{\Omega(n)}}$ of correctly testing PAC learnability for all functions on inputs of length n satisfying the YES or NO conditions. The proof follows from a case analysis: first we show that if given examples drawn from $\mathcal{R}_z$ the testing algorithm queries z with low probability, then with overwhelming probability $C^{\mathcal{O}}$ will classify some functions incorrectly. Then, we show that it is extremely unlikely that $C^{\mathcal{O}}$ can

query z with noticeable probability, because the function $\mathcal{R}_z$ is random and therefore the labeled examples that $C^{\mathcal{O}}$ sees contain no information about z.

Fix n and any oracle learning circuit $C^{(\cdot)}$ of size $s(n) = \text{poly}(n)$, and let $p(n) = \text{poly}(n)$ be an upper bound on the number of labeled examples that $C^{(\cdot)}$ sees. Define

$$S(\mathcal{R}_z) = \{(x_1, \mathcal{R}_z(x_1)), \ldots, (x_{p(n)}, \mathcal{R}_z(x_{p(n)}))\}$$

where the $x_i \leftarrow_{\text{R}} U_n$. Let $C^{\mathcal{O}}(S(\mathcal{R}_z)) = h^{\mathcal{O}}$ be the hypothesis that $C^{\mathcal{O}}$ outputs given labeled examples $S(\mathcal{R}_z)$. Define

$$\mathcal{F}^{\varepsilon,\mathcal{O}}_{\mathsf{far}} = \{\phi : \{0,1\}^n \to \{0,1\} \mid \mathsf{err}((U_n, \phi(U_n), \mathsf{SIZE}^{\mathcal{O}}(n^{\log n})) > \varepsilon\}$$

i.e. the class of all functions that are far from being efficiently computable (even if circuits are allowed $\mathcal{O}$ gates). We write simply $\mathcal{F}^{\varepsilon}_{\mathsf{far}}$ when the oracle $\mathcal{O}$ is clear from context.

Claim 3.3.8. *For* $\varepsilon = 2^{-\log^2 n}$.

$$\Pr_{\mathcal{O}}\left[\bigwedge_{z\in\{0,1\}^n} \{C^{\mathcal{O}} \textit{ accepts } \mathcal{R}_z \textit{ w.h.p.}\} \wedge \bigwedge_{\phi\in\mathcal{F}^{\varepsilon,\mathcal{O}}_{\mathsf{far}}} \{C^{\mathcal{O}} \textit{ rejects } \phi \textit{ w.h.p.}\}\right] \leq 2^{-2^{\Omega(n)}}$$

This claim implies the lemma, since taking a union bound over all $2^{O(s\log(s))}$ circuits of size $s(n) = \text{poly}(n)$ shows that the probability of there existing any circuit that correctly tests consistency is still $2^{-2^{\Omega(n)}}$. By the Borel-Cantelli lemma, this means that with probability 1, no family of circuits tests PAC ε-consistency for F on infinitely many input lengths.

We now prove this claim. Define the following terminology. We say that $C^{\mathcal{O}}$ queries $\mathcal{R}_z$ if it asks $\mathcal{O}$ a formula φ that contains a $\mathcal{R}_z$ gate. Explaining the approach of the proof in some more detail, first we will show that the probability $C^{\mathcal{O}}$ correctly tests PAC ε-consistency without querying $\mathcal{R}_z$ is small because the functions $\mathcal{R}_z$ look

random. Therefore, if $C^{\mathcal{O}}$ were to accept many $\mathcal{R}_z$ without querying z, then it is accepting a random-looking function. But this means it will most likely also accept many random-looking functions, and some of these functions will not be efficiently computable with $\mathcal{O}$ gates, which means it makes a mistake and accepts a function that is far from being efficiently computable. Second, we will show that the probability that $C^{\mathcal{O}}$ queries z given only $p(n)$ queries to the oracle is small because the output of $\mathcal{R}_z$ contains essentially no information about z itself. Define the following events:

- G^{ε}: event over the choice of $\mathcal{O}$ that $C^{\mathcal{O}}$ correctly tests PAC ε-consistency of F.
- A_z: event over the choice of $\mathcal{O}$ that $\Pr_{S(\mathcal{R}_z)}[C^{\mathcal{O}}(S(\mathcal{R}_z)) \text{ accepts } \mathcal{R}_z] \geq 1 - 2^{-n}$.
- B_z: event over the choice of $\mathcal{O}$ that $\Pr_{S(\mathcal{R}_z)}[C^{\mathcal{O}}(S(\mathcal{R}_z)) \text{ queries } \mathcal{R}_z)] > 1/2$

Notice that for all z, $A_z \subseteq G^{\varepsilon}$. We develop the LHS of Claim 3.3.8

$$\Pr_{\mathcal{O}}[G^{\varepsilon}] = \Pr_{\mathcal{O}}\left[G^{\varepsilon} \wedge \bigwedge_{z \in \{0,1\}^n} A_z\right] \tag{3.3.1}$$

$$\leq \Pr_{\mathcal{O}}\left[G^{\varepsilon} \wedge \bigwedge_{z \text{ hard}} A_z\right] \tag{3.3.2}$$

$$\leq \Pr_{\mathcal{O}}\left[G^{\varepsilon} \wedge \bigwedge_{z \text{ hard}} (A_z \vee B_z)\right] \tag{3.3.3}$$

$$\leq \Pr_{\mathcal{O}}\left[G^{\varepsilon} \wedge \left\{\exists z \text{ hard } s.t.\ A_z \wedge \overline{B_z}\right\}\right] + \Pr_{\mathcal{O}}\left[\bigwedge_{z \text{ hard}} B_z\right] \tag{3.3.4}$$

This formalizes our above intuition, since the first term is the probability that $C^{\mathcal{O}}$ tests PAC ε-consistency for F correctly given that for some $z \in \{0,1\}^n$, $C^{\mathcal{O}}$ accepts $\mathcal{R}_z$ without querying z, and the second term is the probability that $C^{\mathcal{O}}(S(\mathcal{R}_z))$ queries $\mathcal{R}_z$ with noticeable probability on every the hard z.

Bounding the first term of Inequality 3.3.4. By taking a union bound, it suffices

to bound

$$\sum_{z \text{ hard}} \Pr_{\mathcal{O}}[G^{\varepsilon} \wedge A_z \wedge \overline{B_z}] = \sum_{z \text{ hard}} \mathbb{E}_{\mathcal{R}'} \Pr_{\mathcal{R}_z}[G^{\varepsilon} \wedge A_z \wedge \overline{B_z} \mid \mathcal{R}'] \tag{3.3.5}$$

where $\mathcal{R}'$ is a fixing of the entire oracle $\mathcal{R}$ *except* for the function $\mathcal{R}_z$, which remains random.

For any function $f : \{0,1\}^n \to \{0,1\}$, let $\mathcal{O}_f$ denote the oracle $\mathcal{O}$ as defined in Definition 3.3.3 except with the oracle $\mathcal{R}$ fixed as follows: $\mathcal{R}_z = f$ and $\mathcal{R}_{z'} = \mathcal{R}'_{z'}$ for all $z' \neq z$. To bound Inequality 3.3.5, we prove the following:

Lemma 3.3.9. *For any oracle $\mathcal{O}_f$ parameterized by a function $f : \{0,1\}^n \to \{0,1\}$, if $G^{\varepsilon} \wedge A_z \wedge \overline{B_z}$ holds for the oracle $\mathcal{O}_f$, then*

$$\Pr_{\phi}[G^{\varepsilon} \wedge A_z \wedge \overline{B_z} \text{ holds for } \mathcal{O}_{\phi}] \leq 2^{-2^{\Omega(n)}}$$

where ϕ is a random function chosen from all functions mapping $\{0,1\}^n \to \{0,1\}$.

First we assume this lemma to conclude the bound on the first term of Inequality 3.3.4. Suppose $\Pr_{\mathcal{R}_z}[G^{\varepsilon} \wedge A_z \wedge \overline{B_z} \mid \mathcal{R}']$ is non-zero (otherwise we are done), and let f be a function such that $\mathcal{O}_f$ satisfies $G^{\varepsilon} \wedge A_z \wedge \overline{B_z}$. Then applying Lemma 3.3.9 says that

$$\Pr_{\mathcal{R}_z}[G^{\varepsilon} \wedge A_z \wedge \overline{B_z} \mid \mathcal{R}'] \leq 2^{-2^{\Omega(n)}}$$

and thus bounds first term of Inequality 3.3.4 by $2^{-2^{\Omega(n)}}$.

Proof of Lemma 3.3.9. By hypothesis, $\mathcal{O}_f$ satisfies:

1. A_z holds, so $\Pr_{S(\mathcal{R}_z)}[C^{\mathcal{O}_f}(S(\mathcal{R}_z)) \text{ accepts } f] > 1 - 2^{-n}$.
2. $\overline{B_z}$ holds, so $\Pr_{S(\mathcal{R}_z)}[C^{\mathcal{O}_f}(S(\mathcal{R}_z)) \text{ queries } z] \leq 1/2$.
3. G^{ε} holds, so for all $\phi \in \mathcal{F}_{\mathsf{far}}^{\varepsilon,\mathcal{O}_f}$, $\Pr_{S(\mathcal{R}_z)}[C^{\mathcal{O}_f}(S(\mathcal{R}_z)) \text{ accepts } \phi] \leq 2^{-n}$.

The high-level argument is as follows: first, a random function ϕ is unlikely to be close to any efficiently computable function. Next, because we want $\overline{B_z}$ to hold, the circuit C fails to distinguish between $\mathcal{O}_f$ and $\mathcal{O}_\phi$ with probability roughly $1/2$. But $C^{\mathcal{O}_f}$ is supposed to reject ϕ while $C^{\mathcal{O}_\phi}$ is supposed to accept ϕ, and therefore this is a contradiction.

More formally, the number of oracle circuits of size $n^{\log\log n}$ is bounded by

$$|\mathsf{SIZE}^{\mathcal{O}_f}(n^{\log\log n})| \leq 2^{O(n^{\log\log n}\log\log n\log n)} = 2^{2^{o(n)}} \tag{3.3.6}$$

For any function ϕ, a Chernoff bound tells us that the number of functions close to ϕ, *i.e.* functions ψ satisfying $\mathsf{err}((U_n, \phi(U_n)), \psi) \leq \varepsilon$, is at most $2^{2^n - \varepsilon^2 2^n/8}$. Therefore, combined with Inequality 3.3.6, this implies that

$$\frac{|\mathcal{F}_{\mathsf{far}}^{\varepsilon,\mathcal{O}_f}|}{2^{2^n}} \geq 1 - |\mathsf{SIZE}^{\mathcal{O}_f}(n^{\log\log n})| \cdot 2^{-\varepsilon^2 2^n/8} = 1 - 2^{-2^{\Omega(n)}} \tag{3.3.7}$$

where the last estimate follows from our choice of $\varepsilon = 2^{-\log^2 n}$. This implies that the probability that a random function lands in $\mathcal{F}_{\mathsf{far}}^{\varepsilon,\mathcal{O}_f}$ is at least $1 - 2^{-2^{\Omega(n)}}$.

Claim 3.3.10. *$\forall \phi \in \mathcal{F}_{\mathsf{far}}^{\varepsilon,\mathcal{O}_f}$ the event $G^\varepsilon \wedge A_z \wedge \overline{B_z}$ does not hold for the oracle $\mathcal{O}_\phi$.*

Suppose for the sake of contradiction that it did hold: A_z means that $C^{\mathcal{O}_\phi}$ accepts ϕ with probability at least $1 - 2^{-n}$ and $\overline{B_z}$ means that $C^{\mathcal{O}_\phi}$ queries z with probability at most $1/2$ over the choice of examples. Unless C queries z, the two oracles $\mathcal{O}_f$ and $\mathcal{O}_\phi$ are identical, therefore $C^{\mathcal{O}_f}$ also accepts ϕ with probability $> 1/2 - 2^{-n}$. But ϕ is a NO instance for the oracle $\mathcal{O}_f$ because $\phi \in \mathcal{F}_{\mathsf{far}}^{\varepsilon,\mathcal{O}_f}$, so $C^{\mathcal{O}_f}$ should reject $\phi \in \mathcal{F}_{\mathsf{far}}^{\varepsilon,\mathcal{O}_f}$ with probability $1 - 2^{-n}$. This is a contradiction, therefore $G^\varepsilon \wedge A_z \wedge \overline{B_z}$ never holds for $\mathcal{O}_\phi$ for any $\phi \in \mathcal{F}_{\mathsf{far}}^{\varepsilon,\mathcal{O}_f}$.

Finally, combining Inequality 3.3.7 with Claim 3.3.10, we conclude that over the random choice of $\mathcal{R}_z$ the event $G^\varepsilon \wedge A_z \wedge \overline{B_z}$ does not hold, namely

$$\Pr_\phi[G^\varepsilon \wedge A_z \wedge \overline{B_z} \text{ holds for } \mathcal{O}_f] \leq 2^{-2^{\Omega(n)}}$$

■

Bounding the second term of Inequality 3.3.4. We will show that if the learner C can query $\mathcal{R}_z$ with noticeable probability given a random set of labeled examples $S(\mathcal{R}_z)$, it can be used to "invert" $\mathcal{R}$ in the following sense: view $\mathcal{R}$ as a function $\{0,1\}^n \to \{0,1\}^{2^n}$ where each input z is mapped to the truth table of $\mathcal{R}_z$. We say that a (computationally unbounded) procedure $A^{\mathcal{R}}$ inverts $\mathcal{R}$ using q queries if for every y in the image of $\mathcal{R}$, we have $A^{\mathcal{R}}(y) = \mathcal{R}^{-1}(y)$ ($\mathcal{R}$ is almost surely injective so we assume it to be the case) and A makes at most q queries to $\mathcal{R}$.

To apply this to our setting, we will show that if $C^{\mathcal{O}}(S(\mathcal{R}_z))$ is able to query $\mathcal{R}_z$ with probability $\geq 1/2$ over $S(\mathcal{R}_z)$, then it can be used to build an inverter for $\mathcal{R}$ making only $O(p(n)n)$ queries. Then we show that with high probability this is impossible.

Lemma 3.3.11. *For every oracle circuit $C^{\mathcal{O}}$, there exists a procedure $A^{\mathcal{R}}$ such that*

$$\Pr_{\mathcal{R}}\left[\bigwedge_{z \text{ hard}} B_z\right](1-2^{-n}) \leq \Pr_{\mathcal{R}}[A^{\mathcal{R}} \text{ inverts } \mathcal{R} \text{ using } O(p(n)n) \text{ queries}]$$

Proof. We first describe a randomized procedure A' for inverting $\mathcal{R}$. A' is defined using the learning circuit C as follows: on every non-zero input $y \in \{0,1\}^{2^n}$ which is the truth table of some function, emulate C $O(n)$ times using independent randomness, answering C's queries to the example oracle and membership oracle using y as the truth table of the hidden labeling function. To answer queries φ that C makes to $\mathcal{O}$, let Z be the set of z such that $\mathcal{R}_z$ appears in φ. For each $z \in Z$ of length n, A' will query $\mathcal{R}$ to get the truth table $\mathcal{R}_z$. Furthermore, A' checks whether $\mathcal{R}_z = y$, and if so it halts and outputs z. For every $z' \in Z$ where $|z'| = n' \neq n$, A' uses independent coin tosses to set $\mathcal{R}_{z'} \equiv 0^{2^{n'}}$ with probability $1 - 2^{-n'/2}$, and $\mathcal{R}_{z'}$ to be a random function mapping $\{0,1\}^{n'} \to \{0,1\}$ with probability $2^{-n'/2}$. Then A' decides the QBF formula φ using these truth tables (A' can do this since it is unbounded). All

these independent runs together query the oracle at most $O(p(n)n)$ times. Because B_z holds for every z, *i.e.* for each z, when trying to learn $\mathcal{R}_z$ the circuit C queries $\mathcal{R}_z$ with probability at least $1/2$, this means with probability $1-(1/2)^{O(n)} \geq 1-2^{-2n}$ at least one of the emulations will query $z = \mathcal{R}^{-1}(y)$, and so A' will find z. Now take a union bound over all possible non-zero inputs $y = \mathcal{R}_z$ of which there are at most 2^n, still with probability $1-2^{-n}$ the random bits used are simultaneously good for all y. This means for any $\mathcal{R}$ where $\bigwedge_{z \text{ hard}} B_z$ holds, A' inverts $\mathcal{R}$ with probability $1-2^{-n}$. This implies

$$\mathbb{E}_{A',\mathcal{R}|\bigwedge_{z \text{ hard}} B_z} \Pr[A' \text{ inverts } \mathcal{R} \text{ using } O(p(n)n) \text{ queries}] \geq 1-2^{-n}$$

By averaging, this means there *exists* a fixing of the random coins of A' (call A' with these fixed coins A) such that for a $1-2^{-n}$ fraction of the $\mathcal{R}$ where $\bigwedge_{z \text{ hard}} B_z$ holds, A inverts $\mathcal{R}$. The lemma follows. ■

The following lemma concludes the proof of the bound on the second term of Inequality 3.3.4.

Lemma 3.3.12. *For any $A^{\mathcal{R}}$, $\Pr_{\mathcal{R}}[A^{\mathcal{R}}$ inverts $\mathcal{R}$ using $O(p(n)n)$ queries$] \leq 2^{-2^{\Omega(n)}}$*

Proof. The proof is a straightforward generalization of Gennaro and Trevisan's proof [GT00] that a random permutation is hard to invert for circuits, extended to the case where the function is not a permutation but is still injective. The idea is that given A, any function that A can invert can be "succinctly described", and therefore there cannot be too many of them.

Fix any oracle procedure $A^{\mathcal{R}}$ making at most $O(p(n)n)$ to $\mathcal{R}$. Let $N = |\{x \mid \mathcal{R}(x) \leftarrow_{\text{R}} U_{2^n}\}|$ denote the number of hard outputs of $\mathcal{R}$; by Chernoff the probability that $N \notin [2^{n/2-1}, 2^{n/2+1}]$ is bounded by $2^{-\Omega(2^{n/2})}$, so in the following we condition on this

event not happening:

$$\Pr_{\mathcal{R}}[A \text{ inverts } \mathcal{R}] \leq 2^{-\Omega(2^{n/2})} \quad + \quad \mathbb{E}_{N \in [2^{n/2-1}, 2^{n/2+1}]} \left[\Pr_{\mathcal{R}}[A \text{ inverts } \mathcal{R} \mid N \text{ hard outputs}] \right]$$

We will further throw out the oracles $\mathcal{R}$ that are not injective (this occurs with probability at most $\leq \binom{N}{2} 2^{-2^n}$). We call $\mathcal{R}$ where neither of these conditions hold "good". Therefore our bound is now:

$$\Pr_{\mathcal{R}}[A \text{ inverts } \mathcal{R}] \leq 2^{-2^{\Omega(n)}} \quad + \quad \mathbb{E}_{N \in [2^{n/2-1}, 2^{n/2+1}]} \left[\Pr_{\mathcal{R} \text{ good}}[A \text{ inverts } \mathcal{R} \mid N \text{ hard outputs}] \right]$$

Notice that with this conditioning, $\mathcal{R}$ is uniform in the set of good $\mathcal{R}$.

To bound the probability on the RHS, we show that A is only capable of inverting very few functions. Here, we follow the argument of [GT00] proving that one-way permutations are hard against circuits.

We give a procedure for describing all possible injective functions $\mathcal{R}$ with N hard outputs as follows: we will keep track of a set $Y \subseteq \{0,1\}^{2^n}$ of "easily describable outputs" y for which we will be able to compute the preimage $x = \mathcal{R}^{-1}(y)$ with very little information using A. For the "hard-to-describe outputs" outside Y we will just explicitly record the function. We show that this is sufficient for reconstructing any $\mathcal{R}$ that A is able to invert. We then prove that the number of functions describable this way is small compared to all possible functions, which gives us the desired bound.

For a fixed $\mathcal{R}$, define Y constructively as follows. Initialize $Y = \varnothing$ and the set $T \subseteq \{0,1\}^{2^n}$ to be the image of the hard instances of $\mathcal{R}$, namely $t \in T$ iff $t = \mathcal{R}(z)$ for some hard instance z. Since we are conditioning on good $\mathcal{R}$, we have that initially $|T| = N$.

Repeatedly perform the following until T is empty: remove the lexicographically first element $t \in T$ and add it to Y. Execute $A^{\mathcal{R}}(t)$ and record the queries $x_1, \ldots, x_m$ (in the order that A makes them) that A makes to $\mathcal{R}$, where $m = O(p(n)n)$. If none

of the x_i satisfy $\mathcal{R}(x_i) = t$, then remove all of the $x_1, \ldots, x_m$ from T. If some x_i satisfies $\mathcal{R}(x_i) = y$, then remove $x_1, \ldots, x_{i-1}$ from T. Repeat by removing the next lexicographically first element of T, adding it to Y, etc.

Clearly we have that $|Y| \geq N/m$. We claim that given the set of hard instances $Z = \mathcal{R}^{-1}(T) \subseteq \{0,1\}^n$ (which is of size N), the set Y, the preimage of Y which we call $X = \mathcal{R}^{-1}(Y) \subseteq Z$, and the explicit values of $\mathcal{R}$ on all inputs $x \in Z \setminus X$, we can completely reconstruct $\mathcal{R}$ as follows. For each $x \notin Z$, $\mathcal{R}(x) = 0^{2^n}$. For each $x \in Z \setminus X$, output the explicitly recorded value. It only remains to match the elements of Y with their correct preimage in X. For each $y \in Y$ in lexicographic order, run $A^{\mathcal{R}}(y)$. The queries $A^{\mathcal{R}}(y)$ makes to $\mathcal{R}$ will all either be for $x \notin X$ in which case we know the answer explicitly, for $x \in X$ such that $\mathcal{R}(x)$ is lexicographically smaller than y and so we already computed the answer previously, or for some $x \in X$ we have not seen in a previous computation, which by construction must mean $x = \mathcal{R}^{-1}(y)$. Either way, we obtain the value $\mathcal{R}^{-1}(y)$.

The number of functions describable in this way is exactly

$$\binom{2^n}{N}\binom{N}{|Y|}\binom{2^{2^n}}{|Y|} \cdot \frac{(2^{2^n} - |Y|)!}{(2^{2^n} - N)!}$$

where the first factor is the number of ways of choosing N hard instances, the second is the choice of X, the third is the choice of Y, and the final is the number of ways of explicitly defining the function on $Z \setminus X$ assuming the function is injective. Therefore, the probability over $\mathcal{R}$ that A inverts $\mathcal{R}$ is exactly the above quantity divided by the

total number of good $\mathcal{R}$, namely $\binom{2^n}{N}\frac{(2^{2^n})!}{(2^{2^n}-N)!}$. So we can calculate that:

$$\Pr_{\mathcal{R}\text{ injective}}[A \text{ inverts } \mathcal{R} \text{ everywhere} \mid N \text{ hard instances}] \leq \frac{\binom{2^n}{N}\binom{N}{|Y|}\binom{2^{2^n}}{|Y|}\cdot\frac{(2^{2^n}-|Y|)!}{(2^{2^n}-N)!}}{\binom{2^n}{N}\frac{(2^{2^n})!}{(2^{2^n}-N)!}} \tag{3.3.8}$$

$$= \frac{\binom{N}{|Y|}}{|Y|!} \tag{3.3.9}$$

$$\leq \left(\frac{N3e}{|Y|^2}\right)^{|Y|} \tag{3.3.10}$$

which is $2^{-2^{\Omega(n)}}$ for $N \leq 2^{n/2+1}$ and $|Y| > N/m = 2^{(1-o(1))n/2}$. ■

This concludes the proof of Lemma 3.3.5. ■

3.3.2 AIOWF do not exist relative to $\mathcal{O}$

Proof of Lemma 3.3.6. The inverter I works as follows: it finds all the z such that $f^{\mathcal{O}}(x)$ queries $\mathcal{R}_z$ with noticeable probability over choice of random input x; call this set H the "heavy" queries. We show that by finding H, I knows most of the *hard instances* z such that $f^{\mathcal{O}}$ queries $\mathcal{R}_z$. Let $\mathcal{O}'$ be the oracle defined exactly as $\mathcal{O}$ except that all queries to $\mathcal{R}_{z'}$ for instances $z' \notin H$ are answered with 0. With knowledge of H and access to $\mathcal{O}$ the inverter I can simulate a $\mathcal{O}'$ oracle. It follows from standard results (for example Proposition A.1.2) that because $|H| = \text{poly}(n)$, the $\mathcal{O}'$ oracle can be used to invert $\mathcal{O}'$ computations.

We could try to use this to invert $f^{\mathcal{O}}$, but the computation of $f^{\mathcal{O}}(x)$ may query hard instances outside H, and so $f^{\mathcal{O}}(x) \neq f^{\mathcal{O}'}(x)$ for some x. However, we argue that, by the definition of heavy and because hard instances are scattered at random, the probability over a *random* x that the computation $f^{\mathcal{O}}(x)$ queries a hard instance outside H cannot be too high. Therefore, the distributions $(x, f^{\mathcal{O}}(x))$ and $(x, f^{\mathcal{O}'}(x))$

for $x \leftarrow_{\mathrm{R}} U_n$ are statistically close, and so the inverter using $\mathcal{O}'$ to invert $f^{\mathcal{O}'}$ can also invert $f^{\mathcal{O}}$ with almost as good probability. That is, if I inverts y pretending that it is the output of $f^{\mathcal{O}'}$, then with high probability over random y the inverter produces $x \in (f^{\mathcal{O}})^{-1}(y)$.

We proceed now formally. We describe and analyze an algorithm I that with probability 2^{-s} over the choice of oracle, inverts all f of computable by a circuit of size s. This proves the lemma, since by the Borel-Cantelli lemma this means $I^{\mathcal{O}}$ inverts all except finitely many circuits with probability 1 over $\mathcal{O}$.

Let f be any function computable by an circuit $C^{\mathcal{O}}$ with $\mathcal{O}$ gates of size s, where f takes inputs of length n. Let $g_1, \ldots, g_s$ be the oracle gates of $C^{\mathcal{O}}$ in topologically sorted order. Let D be the (efficient) circuit taking inputs φ a $\mathsf{QBF}^{\mathcal{R}}_*$ formula and $z \in \{0,1\}^*$ and outputting 1 if φ contains a $\mathcal{R}_z$ gate, and outputs 0 otherwise.

Set the heaviness threshold to be $\alpha = 100s^8$. In sorted order, I finds all z such that $C^{\mathcal{O}}(U_n)$ queries $\mathcal{O}$ with a formula containing a $\mathcal{R}_z$ gate with probability larger than $1/\alpha$ using the following procedure.

First, I initializes the set $Z_0 = \{z \mid |z| \leq 8 \log s\}$. Then, to construct Z_i, the set of heavy queries up till the i'th query, using Z_{i-1}, I does the following. Let the circuit Q'_i be the sub-circuit of C that computes queries for g_i. We transform Q'_i into a related circuit Q_i by replacing each oracle gate g_j, $j < i$ that appears in Q'_i (these are the only oracle gates that g_i depends on since we work in sorted order) with the following modification: on input φ, replace each $\mathcal{R}_z$ gate inside φ where $z \notin Z_j$ by a constant 0 gate, and then call $\mathcal{O}$ with this modified formula. This transformation forces all the hard instances that φ queries to be in Z_j.

Note that $Q_i(x) = \varphi$ is exactly saying that $C(x)$ queries φ at g_i, conditioned on each previous oracle gate g_j only being queried on heavy instances ($z \in Z_j$) or

easy instances ($\mathcal{R}_z \equiv 0$). Since Q_i only makes oracle queries containing $\mathcal{R}_z$ gates for $z \in Z_{i-1}$, this means Q_i is computable using only a $\mathbf{PSPACE}^{\mathcal{R}'_{i-1}}$ oracle (*i.e.* it does not need a $\mathbf{PSPACE}^{\mathcal{R}}_*$ oracle), where $\mathcal{R}'_{i-1}(z,x) = \mathcal{R}(z,x)$ for $z \in Z_{i-1}$ and is zero otherwise. Since I knows Z_{i-1}, it can simulate a $\mathbf{PSPACE}^{\mathcal{R}'_{i-1}}$ oracle. **PSPACE** oracles are able to compute the probabilities in the output distribution of **PSPACE** computations and this relativizes, for example as stated in Proposition A.1.3. We invoke the algorithm given by this proposition on input $(Q_i, D, 1^\alpha)$ to get $\{z \mid \Pr[Q_i(x) = \varphi \wedge D(\varphi, z) = 1] > 1/\alpha\}$, which we add to Z_{i-1} to obtain Z_i.

Proposition A.1.3 guarantees Z_s is the collection of all z such that there exists i such that Q_i queries z with probability $> 1/\alpha$ over the choice of random input x. This set Z_s is our set of heavy elements.

We now show that with high probability over $\mathcal{O}$, if I knows Z_s then it knows most of the hard instances that $f^{\mathcal{O}}$ might have queried, and so it can invert $f^{\mathcal{O}}$ almost everywhere. Formally, let $B(x)$ be the bad event that $f^{\mathcal{O}}(x)$ queries some hard z outside Z_s.

Claim 3.3.13.

$$\Pr_{\mathcal{R}}\left[\Pr_{x \leftarrow_R U_n}[B(x)] > \tfrac{1}{s}\right] \leq 2^{-s^2}$$

First we use this claim to prove the lemma: by a union bound over all f computable by size s circuits, of which there are at most $2^{O(s \log s)}$, this means that for a $1 - 2^{-s^2+O(s\log s)} \geq 1 - 2^{-s}$ fraction of the $\mathcal{R}$ that with probability $1 - 1/s$ over x, $f^{\mathcal{O}}(x)$ never queries hard $z \notin Z_s$. But in this case we can give f oracle access to $\mathbf{PSPACE}^{\mathcal{R}'_s}$ instead of $\mathcal{O}$ and get the same output, where $\mathcal{R}'_s$ is just $\mathcal{R}'_i$ as defined above with $i = s$. This implies $\Delta\left((x, f^{\mathcal{O}}(x)), (x, f^{\mathbf{PSPACE}^{\mathcal{R}'_s}}(x))\right) \leq 1/s$.

Furthermore, **PSPACE** oracle can invert **PSPACE** computations and this relativizes, for example in Proposition A.1.2. (Note that this does not imply that

$\mathbf{PSPACE}^{\mathcal{R}}_*$ can invert $\mathbf{PSPACE}^{\mathcal{R}}_*$ because the polynomial-space machine in the definition of $\mathbf{PSPACE}^{\mathcal{R}}_*$ does not have unhindered access to its oracle.)

I knows Z_s so it can use Z_s and $\mathcal{O}$ to simulate $\mathbf{PSPACE}^{\mathcal{R}'_s}$, so it can use Proposition A.1.2 to compute uniformly random preimages of $f^{\mathbf{PSPACE}^{\mathcal{R}'_s}}$ with failure probability 2^{-m}, giving us

$$\Delta\left((x, f^{\mathbf{PSPACE}^{\mathcal{R}'_s}}(x)), \quad (I^{\mathcal{O}}(y), y \mid y = f^{\mathbf{PSPACE}^{\mathcal{R}'_s}}(x))\right) \leq 2^{-m}$$

Putting these together by the triangle inequality, we have

$$\Delta((x, f^{\mathcal{O}}(x)), (I^{\mathcal{O}}(y), y \mid y = f^{\mathcal{O}}(x))) \leq 2/s + 2^{-m}$$

which proves the lemma modulo Claim 3.3.13. In fact, we prove something much better: $I^{\mathcal{O}}$ actually gives an almost uniformly random preimage of y.

It remains to prove Claim 3.3.13. Define inductively $B_i(x)$ as the event that $f^{\mathcal{O}}(x)$ queries a hard $z \notin Z_i$ in the i'th query but all prior queries j are either easy or in Z_j. Since $Z_i \subseteq Z_{i+1}$, we have that $B(x) \subseteq \bigcup_{i=1}^{s} B_i(x)$. By averaging:

$$\begin{aligned}
\Pr_{\mathcal{R}}\left[\Pr_x[B(x)] > \tfrac{1}{s}\right] &\leq \Pr_{\mathcal{R}}\left[\Pr_x\left[\bigcup_{i=1}^{s} B_i(x)\right] > \tfrac{1}{s}\right] \\
&\leq \Pr_{\mathcal{R}}\left[\exists i,\ \Pr_x[B_i(x)] > \tfrac{1}{s^2}\right] \\
&\leq \sum_{i=1}^{s} \Pr_{\mathcal{R}}\left[\Pr_x[B_i(x)] > \tfrac{1}{s^2}\right]
\end{aligned}$$

We claim that for each i, $\Pr_{\mathcal{R}}[\Pr_x[B_i(x)] > 1/s^2] \leq 2^{-2s^2}$, which we prove using a case analysis. Showing this concludes the proof of the lemma since $s2^{-2s^2} \leq 2^{-s^2}$.

The case analysis roughly goes as follows: either the probability that $f^{\mathcal{O}}$ makes a light i'th query (*i.e.* a query not in Z_i) is small, in which case the probability it makes a light and hard query is also small, or the probability that $f^{\mathcal{O}}$ makes a light i'th query is large, in which case there are many distinct light queries, and since each light query

is hard independently with probability $\leq 1/s^4$, we can show that it is unlikely over the choice of oracle that a $1/s^2$ fraction of these light queries are hard.

Formally, let $\mathsf{NotInZ}_i(x)$ be the event that $f^{\mathcal{O}}$'s i'th query is not in Z_i conditioned on all queries $j < i$ being either in Z_j or easy. (The only difference between NotInZ_i and B_i is that in B_i we also demand the i'th query be hard.) We have that

$$\Pr_{\mathcal{R}}[\Pr_x[B_i(x)] > 1/s^2] = \Pr_{\mathcal{R}}\left[\left\{\Pr_x[B_i(x)] > 1/s^2\right\} \wedge \left\{\Pr_x[\mathsf{NotInZ}_i(x)] \geq 1/s^2\right\}\right] + \Pr_{\mathcal{R}}\left[\left\{\Pr_x[B_i(x)] > 1/s^2\right\} \wedge \left\{\Pr_x[\mathsf{NotInZ}_i(x)] < 1/s^2\right\}\right]$$

Clearly the second term is 0 because $B_i(x) \subseteq \mathsf{NotInZ}_i(x)$.

To bound the first term, we inductively fix $\mathcal{R}$ up until the i'th query as follows: let $\mathcal{R}_0$ be a fixing of all $\mathcal{R}_z$ with $z \in Z_0$. Let Z_i be the set of heavy queries conditioned on $\mathcal{R}_{i-1}$ and the event that $f^{\mathcal{O}}(x)$'s first $i-1$ queries are either easy or in Z_{i-1}, and let $\mathcal{R}_i$ be a fixing of $\mathcal{R}_z$ with $z \in Z_i$ conditioned on $\mathcal{R}_{i-1}$. Thus, we can write:

$$\begin{aligned}
&\Pr_{\mathcal{R}}\left[\left\{\Pr_x[B_i(x)] > 1/s^2\right\} \wedge \left\{\Pr_x[\mathsf{NotInZ}_i(x)] \geq 1/s^2\right\}\right] \\
&= \mathbb{E}_{\mathcal{R}_{i-1}} \Pr_{\mathcal{R}}\left[\left\{\Pr_x[B_i(x)] > 1/s^2\right\} \wedge \left\{\Pr_x[\mathsf{NotInZ}_i(x)] \geq 1/s^2\right\} \mid \mathcal{R}_{i-1}\right] \\
&\leq \mathbb{E}_{\mathcal{R}_{i-1}} \Pr_{\mathcal{R}}\left[\left\{\Pr_x[B_i(x) \mid \mathsf{NotInZ}_i(x)] > 1/s^2\right\} \mid \left\{\Pr_x[\mathsf{NotInZ}_i(x)] \geq 1/s^2\right\} \wedge \mathcal{R}_{i-1}\right]
\end{aligned}$$

where in the last line we used the fact that $B_i(x) \subseteq \mathsf{NotInZ}_i(x)$. For each such fixing of $\mathcal{R}_{i-1}$, since the probability that the i'th query is light is at least $1/s^2$, the probability that a specific light z is asked as the i'th query conditioned on $\mathsf{NotInZ}_i(x)$ is at most $s^2/\alpha = 1/(100s^6)$. Each i'th query is hard independently with probability at most $1/s^4$ over the choice of oracle (because Z_0 contains all queries of length up to $8 \log s$, the oracle is random only on longer inputs). If each light query were asked with probability *exactly* $1/(100s^6)$ then we could apply a Chernoff bound, which says that the probability that more than $1/s^2$ of the light queries are hard given that each light query is hard with probability $1/s^4$ is at most $2^{-100s^6/(4s^4))} \leq 2^{-2s^2}$. By a simple

generalization of the Chernoff bound stated in Lemma A.2.1, we can show that the same bound holds even though we are guaranteed that each light query is asked with probability *at most* $1/(100s^6)$, so this concludes the proof of the lemma. ■

3.3.3 ∀∃-construction-black-box reductions

As we defined in Definition 1.3.3, one can consider a broader notion of reducibility called ∀∃-construction-black-box reductions. The proof of Theorem 3.3.1 actually is strong enough to rule out such reductions as well.

Theorem 3.3.14. *There exists no ∀∃-construction-black-box reduction such that, given a concept class F such that testing PAC consistency of F over the uniform distribution with a membership oracle is hard, constructs AIOWF.*

Proof. The proof of Theorem 3.3.1 in fact already rules out ∀∃-construction-black-box reductions. To rule out relativizing reductions that uses a primitive P to build a primitive Q, it sufficed to construct an oracle relative to which P exists but Q does not. This is what we did in the proof of Theorem 3.3.1: we showed that relative to the oracle $\mathcal{O}$ that decides $\mathsf{QBF}_*^{\mathcal{R}}$, it is hard to test PAC consistency of $F = \{\mathcal{R}_z\}_{z\in\{0,1\}^n}$ over the uniform distribution with a membership oracle, but AIOWF do not exist.

To rule out ∀∃ construction-black-box reductions, we must exhibit a concept class (given as an oracle) for which testing PAC consistency is hard, such that no matter how one tries to build an AIOWF using this concept class, there is an efficient adversary (also given the same concept class as an oracle) that inverts the AIOWF. We cannot use simply $F = \{\mathcal{R}_z\}_{z\in\{0,1\}^n}$ because oracle access to F is not sufficient to invert AIOWF. However, observe that it is also hard to test PAC consistency of the class of functions computable by $\mathsf{QBF}_*^{\mathcal{R}}$ formulas (since in particular it contains

$\{\mathcal{R}_z\}$), and these functions are efficiently computable given the oracle $\mathcal{O}$. Also, access to a $\mathsf{QBF}_*^{\mathcal{R}}$ oracle is sufficient to invert AIOWF. This implies the theorem. ■

3.4 CircCons and CircLearn: efficient example oracles

The PAC model gives only oracle access to the distribution of labeled examples. In particular, the learning algorithm has no idea how the distribution is generated, and it is conceivable that the distribution is not even efficiently samplable. In order to better highlight this distinction, which will be useful later on, we formalize here the problem of learning when the distribution of labeled examples is efficiently samplable, and where the learning algorithm gets to see the circuit that samples this distribution. We emphasize that this problem is not trivial: just because the learning algorithm can see the circuit C that generates the distribution (X, Y), it only knows how to sample (X, Y) together and does not necessarily know how to efficiently compute the label Y given just X as input. In fact in general C may sample a distribution where X, Y are independent and such a labeling does not exist.

Definition 3.4.1 ($\mathsf{CircCons}^F_{\alpha,\beta}$)**.** (α, β)-circuit consistency for F, denoted $\mathsf{CircCons}^F_{\alpha,\beta}$, is a promise problem where an input is a circuit C of size s sampling a joint distribution (X, Y) where X is over $\{0,1\}^n$ and Y is over $\{0,1\}$ and $n > \sqrt{s}$.

- YES instance: $\mathsf{err}((X, Y), F) < \alpha$
- NO instance: $\mathsf{err}((X, Y), \mathsf{SIZE}(n^{\log\log n})) > \beta$

The condition $n > \sqrt{s}$ is simply to assure that n is polynomially related to s and can replaced with the condition $n > s^{\varepsilon}$ for any $\varepsilon > 0$ without qualitatively affecting any of our results.

We say that CircCons is hard if there exist any $0 \leq \alpha < \beta \leq 1$ satisfying $\beta - \alpha \geq 1/\text{poly}(n)$ such that $\mathsf{CircCons}_{\alpha,\beta}^{\mathsf{SIZE}(n^2)} \notin \mathbf{BPP}$.

Definition 3.4.2. A procedure A circuit-learns a concept class F, or A solves $\mathsf{CircLearn}_{\varepsilon}^{F}$, if on input circuit C of size s computing a distribution (X, Y) over $\{0,1\}^{n+1}$ where $n > \sqrt{s}$ and an accuracy parameter ε, A outputs a hypothesis h such that

$$\mathsf{err}((X,Y),h) \leq \mathsf{err}((X,Y),F) + \varepsilon$$

As with CircCons (Definition 3.4.1), the condition $n > \sqrt{s}$ is simply to assure that n is polynomially related to s and can replaced with the condition $n > s^{\varepsilon}$ for any $\varepsilon > 0$ without qualitatively affecting any of our results.

We say that $\mathsf{CircLearn}^{F}$ is hard against uniform (resp. non-uniform) algorithms if there exists some $\varepsilon = 1/\text{poly}(n)$ such that no uniform (resp. non-uniform) A running in time $\text{poly}(n)$ solves $\mathsf{CircLearn}_{\varepsilon}^{F}$. We say that CircLearn is hard if $\mathsf{CircLearn}^{\mathsf{SIZE}(n^2)}$ is hard.

The following propositions follows easily from the definitions:

Proposition 3.4.3. *If* $\mathsf{CircCons}_{0,\beta}$ *is hard for any* $\beta \geq 1/\text{poly}(n)$ *then testing PAC consistency is hard.*

Proposition 3.4.4. *If* CircCons *is hard, then* CircLearn *is hard.*

3.5 CircLearn and AIOWF

In Section 3.3 and Section 3.3.3, we showed that a wide class of techniques cannot prove that the ability to invert AIOWF implies the ability to test PAC consistency. The main observation we used was that in testing PAC consistency the tester only gets to observe the labeled examples and does not know how they were generated. It

turns out that by changing the model so that the tester knows how the examples are generated (namely, looking at the problem CircCons or CircLearn instead of testing PAC consistency or standard PAC learning), we *can* prove that that the ability to invert AIOWF implies the ability to solve say CircLearn.

Given a circuit C sampling a distribution (X, Y), let C_1 be the subcircuit of C that outputs only the first n bits of the output of C (corresponding to X), and let C_2 be the subcircuit that outputs the last bit of C (corresponding to Y). Intuitively, the following lemma says that there is an efficient algorithm that, given input C and access to an inverter I for C_1, constructs a hypothesis h that labels (X, Y) almost as good as information-theoretically possible.

Lemma 3.5.1 (Agnostically learning circuits)**.** *There exists an algorithm A that takes input C a circuit of size s sampling a distribution (X, Y), an accuracy parameter ε, and oracle access to I satisfying the following. Suppose I distributionally inverts C_1, namely*

$$\Delta((r, C_1(r)), (I(x), x \mid x = C_1(r))) \leq O(\varepsilon^6/s) \tag{3.5.1}$$

where r is uniformly random. Then A outputs with probability $1 - 2^{-s}$ a hypothesis h that uses oracle access to I such that

$$\mathsf{err}((X, Y), h) \leq \mathsf{err}((X, Y), \mathcal{F}_{\mathsf{all}}) + \varepsilon$$

where $\mathcal{F}_{\mathsf{all}}$ is the set of all (possibly inefficient) functions. Furthermore, both A and h make at most $\mathrm{poly}(s/\varepsilon)$ *non-adaptive queries to I and run in time* $\mathrm{poly}(s/\varepsilon)$.

Proof of Lemma 3.5.1. Consider the function $g(x) = b$ where $\Pr[b = Y \mid X = x] \geq 1/2$ (breaking any ties arbitrarily). Define the *certainty* $\mathsf{cert}(x) = \Pr[g(x) = Y \mid X = x]$ (for example, $\mathsf{cert}(x) = 1$ if the x completely determines Y, and $\mathsf{cert}(x) = 1/2$ if Y is completely random given x). The function g maximizes agreement with (X, Y)

because for any function f and any x, we have that

$$\Pr[f(x) \neq Y \mid X = x] \geq 1 - \mathsf{cert}(x) = \Pr[g(x) \neq Y \mid X = x]$$

and so therefore for all $f \in \mathcal{F}_{\mathsf{all}}$,

$$\begin{aligned}
\mathsf{err}((X,Y), f) &= \Pr[f(X) \neq Y] \\
&= \mathbb{E}_{x \leftarrow_{\mathrm{R}} X} \Pr[f(x) \neq Y \mid X = x] \\
&\geq \mathbb{E}_x(1 - \mathsf{cert}(x)) = \Pr[g(X) \neq Y] \\
&= \mathsf{err}((X,Y), g)
\end{aligned}$$

Thus, we call g the optimal labeling since it satisfies $\mathsf{err}((X,Y), g) = \mathsf{err}((X,Y), \mathcal{F}_{\mathsf{all}})$, and our goal will be to show that if AIOWF do not exist, then we can label efficiently as well as g does plus some additional error ε.

We start by exhibiting a randomized hypothesis that performs well, and then show it can be derandomized.

Randomized hypothesis: let $h(\omega, x)$ be the randomized hypothesis that takes random bits $\omega = (\omega_1, \ldots, \omega_m)$ where $m = \Theta(s/\varepsilon^4)$ and each ω_i can be used to run the inverter I. On input x, h does the following:

1. Obtain $r_i = I(x; \omega_i)$ with accuracy parameter $\frac{\varepsilon^2}{2m} = O(\frac{\varepsilon^6}{s})$ for $i = 1$ to m, each time using independent coins ω_i.
2. Compute $y_i = Y(r_i)$ for all the i.
3. Output $\mathsf{Majority}(y_1, \ldots, y_m)$.

Clearly h runs in time $\mathrm{poly}(s/\varepsilon)$ as long as I also does.

Analyzing randomized hypothesis: to analyze h, we first show that an "ideal" hypothesis would do well and then show that h is close to being ideal. An ideal

hypothesis h_0 is defined as h, except that it calls an ideal inverter I_0, which satisfies:

$$(r, C_1(r)) = (I_0(x), x \mid x = C_1(r)) \tag{3.5.2}$$

where the randomness is over uniform r and the internal randomness of I_0. That is, the ideal inverter I_0 computes exactly the conditional distribution on preimages r given an output $x = C_1(r)$. In particular, this means that for every x and for every y_i that h_0 generates, it holds that $\Pr_{I_0}[g(x) = y_i \mid y_i = Y(I_0(x))] = \mathsf{cert}(x)$. We claim that the majority of the y_i is rarely wrong, namely for every x in the support of X,

$$\Pr_{h_0}[h_0(x) = \mathsf{Majority}(y_1, \ldots, y_m) \neq Y \mid X = x] \leq \max_{b \in \{0,1\}} \Pr[b = Y \mid X = x] + \varepsilon^2/2 \tag{3.5.3}$$

We prove Inequality 3.5.3 by a case analysis:

1. $\mathsf{cert}(x) > \frac{1}{2} + \varepsilon^2/4$: in this case the function $g(x)$ clearly gives the "right answer" so we want to show that $h_0(x) = g(x)$ with high probability. Let Z_i denote the random variable that is 1 if $g(x) = y_i$ and 0 otherwise. By definition, $\Pr[Z_i = 1] = \Pr[g(x) = y_i] = \mathsf{cert}(x)$ and therefore

$$\Pr[h_0(x) = \mathsf{Majority}(y_1, \ldots, y_i) \neq g(x)] = \Pr\left[\frac{1}{m}\sum_{i=1}^{m} Z_i < \frac{1}{2}\right] \leq 2^{-\varepsilon^4 m/32}$$

where the last inequality follows from a Chernoff bound. Since $m = \Theta(s/\varepsilon^4)$, this means that $\Pr[h_0(x) \neq g(x)] \leq 2^{-s}$. Finally, we obtain:

$$\begin{aligned}\Pr[h_0(x) \neq Y \mid X = x] &\leq \Pr[g(x) \neq Y \mid X = x] \\ &\quad + \Pr[h_0(x) \neq g(x)] \Pr[g(x) = Y \mid X = x] \\ &\leq \max_{b \in \{0,1\}} \Pr[b \neq Y \mid X = x] + 2^{-s}\end{aligned}$$

where the last line holds because $g(x) = b$ the bit minimizing $\Pr[b \neq Y \mid X = x]$.

2. $\mathsf{cert}(x) \leq \frac{1}{2} + \varepsilon^2/4$: in this case there is little certainty given x what Y should be, and even the "optimal" function g is frequently wrong. This means that although h_0 may disagree with g frequently, still it is almost as good, namely $\Pr[h_0(x) = Y \mid X = x] \leq \frac{1}{2} + \varepsilon^2/4$ and for both $b \in \{0,1\}$, it holds that $\Pr[b = Y \mid X = x] \geq \frac{1}{2} - \varepsilon^2/4$. Therefore

$$\begin{aligned}\Pr[h_0(x) = Y \mid X = x] &\leq \tfrac{1}{2} + \varepsilon^2/4 \\ &\leq \max_{b \in \{0,1\}} \Pr[b = Y \mid X = x] + \varepsilon^2/2\end{aligned}$$

Since Inequality 3.5.3 holds for all x, we clearly have for random X that

$$\Pr_{X,Y}[h_0(X) \neq Y] \leq \Pr[g(X) \neq Y] + \varepsilon^2/2$$

Inequality 3.5.1 and Equation 3.5.2 imply that I and I_0 behave almost the same:

$$\Delta((I(X), X), \quad (I_0(X), X)) \leq \tfrac{\varepsilon^2}{2m}$$

Since the randomized hypothesis h and ideal hypothesis h_0 differ only in the fact that the former calls I while the latter calls I_0, and because there are only m calls to the inverters, we can apply the triangle inequality to see that

$$\Delta((X, h_0(X)), (X, h(X))) \leq \varepsilon^2/2$$

Since $\Pr[h_0(X) \neq Y] = \Delta((X,Y), (X, h_0(X)))$ (and similarly for h), this implies that

$$\begin{aligned}\mathsf{err}((X,Y), h) &= \Pr_{h,X,Y}[h(X) \neq Y] && (3.5.4)\\ &\leq \Pr[h_0(X) \neq Y] + \varepsilon^2/2 && (3.5.5)\\ &\leq \Pr[g(X) \neq Y] + \varepsilon^2 && (3.5.6)\\ &= \mathsf{err}((X,Y), \mathcal{F}_{\mathsf{all}}) + \varepsilon^2 && (3.5.7)\end{aligned}$$

Deterministic hypothesis: Inequality 3.5.7 proves that h is a good randomized hypothesis. To derandomize pick the random coins ω for h and output the deterministic hypothesis $h_\omega = h(\omega, \cdot)$. We say that h_ω is good if $\Pr[h_\omega(X) \neq Y] \leq \Pr[g(X) \neq Y] + \varepsilon$ and it is bad otherwise. We claim that $\Pr_\omega[h_\omega \text{ is bad}] \leq \varepsilon$, since otherwise

$$\begin{aligned}
\Pr_{h,X,Y}[h(X) \neq Y] &= \Pr_{\omega,X,Y}[h_\omega(X) \neq Y] \\
&= (1 - \Pr_\omega[h_\omega \text{ is bad}]) \Pr_{X,Y}[h_\omega(X) \neq Y \mid h_\omega \text{ is good}] \\
&\quad + \Pr_\omega[h_\omega \text{ is bad}] \Pr_{X,Y}[h_\omega(X) \neq Y \mid h_\omega \text{ is bad}] \\
&> (1 - \Pr_\omega[h_\omega \text{ is bad}]) \Pr_{X,Y}[g(X) \neq Y] \\
&\quad + \Pr_\omega[h_\omega \text{ is bad}](\Pr_{X,Y}[g(X) \neq Y] + \varepsilon) \\
&> \Pr[g(X) \neq Y] + \varepsilon^2
\end{aligned}$$

which is a contradiction. Here, in the penultimate inequality we used the fact that g is optimal, *i.e.* that $\Pr_{X,Y}[h_\omega(X) \neq Y] \geq \Pr[g(X) \neq Y]$ regardless of whether h_ω is good or bad.

Therefore, we have that

$$\Pr_\omega\left[\left\{\Pr_{X,Y}[h_\omega(X) \neq Y] \leq \Pr_{X,Y}[g(X) \neq Y] + \varepsilon\right\}\right] > 1 - \varepsilon \tag{3.5.8}$$

Using Proposition 2.2.8 this can be repeated many times in order to increase the success probability of obtaining a good hypothesis from $1 - \varepsilon$ to $1 - 2^{-s}$.

Finally, observe that the queries h makes are non-adaptive since they are all computed before seeing any responses from I.

■

We obtain the following corollary:

Corollary 3.5.2. *There is a non-adaptive efficient-oracle reduction that uses an AIOWF inverter and solves* CircLearn.

Proof. One can use non-adaptive access to a AIOWF inverter and construct a AID-OWF inverter (Theorem 2.3.5, Remark 2.3.9). Therefore, suppose that we have instead an oracle I such that for every circuit C_1 of size s, it holds that

$$((r, C_1(r)), (I(C_1, x; \omega), x \mid x = C_1(r))) \leq O(\varepsilon^6/s)$$

where r and ω are independent uniform strings. Furthermore, I runs in time $\text{poly}(s/\varepsilon)$.

Let A be the algorithm given by Lemma 3.5.1. Let A' perform the following: on input a circuit C sampling (X, Y), execute A and respond to queries to invert y using random coins ω by computing $x = I(C, y; \omega)$. This results with probability $1 - 2^{-s}$ in a hypothesis h satisfying

$$\mathsf{err}((X, Y), h) \leq \mathsf{err}((X, Y), \mathcal{F}_{\mathsf{all}}) + \varepsilon \leq \mathsf{err}((X, Y), \mathsf{SIZE}(n^2)) + \varepsilon$$

This solves $\mathsf{CircLearn}_{\varepsilon}^{\mathsf{SIZE}(n^2)}$ in time $\text{poly}(s/\varepsilon)$ and only uses non-adaptive access to I. It is efficient-oracle because the code of the hypothesis h contains the code of the inverter I. ■

Relationship to Universal Extrapolation [IL90]

Lemma 3.5.1 is related in spirit to the work of Impagliazzo and Levin [IL90] on universal extrapolation. One can interpret their work as similarly computing the label of an efficiently samplable distribution. The main difference between our setting and theirs is that we are concerned with the *non-uniform* setting of distributions sampled by non-uniform circuits, while they studied distributions samplable by efficient uniform Turing machines.

3.6 Summary

Figure 3.6 depicts the relationships we have established in this chapter. Notice that several of the implications given by following the arrows multiple hops can also be proven via trivial reductions; for example the fact that CircLearn is hard implies that agnostic learning is hard can be proven via the chain of reductions in Figure 3.6, but it can also be proven trivially since any agnostic learning algorithm can be used to solve CircLearn by using the sampling circuit given as an instance of CircLearn in order to generate labeled examples to pass to the agnostic learning algorithm.

The two messages to take away from this chapter are:

1. Learning or even testing consistency in the standard oracle model is "harder" than inverting AIOWF, at least if one is restricted to standard techniques.

2. Learning or testing consistency when the example oracle is given as a circuit is "easier" than inverting AIOWF.

Open questions: is the hardness of CircCons or CircLearn equivalent to the existence of AIOWF? Or can one prove a separation between them?

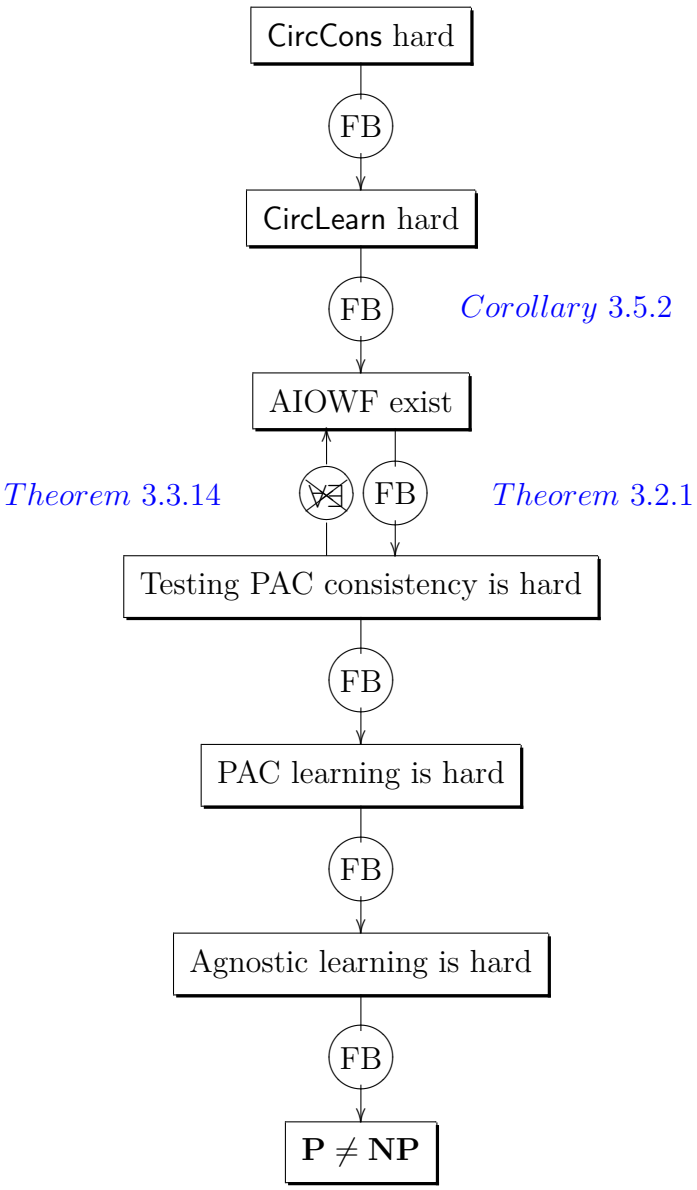

Figure 3.1: Relationship of AIOWF and learning

Chapter 4

Learning and ZK

Zero knowledge proofs were introduced by Goldwasser *et al.* [GMR85] to intuitively capture the notion of "learning nothing about a statement except that it is true". As defined in Section 2.4, they introduce the notion of simulating the view of the verifier, and require that for all efficient verifiers, there exists an efficient simulator such that the "view" generated by the verifier interacting with the prover and the "view" generated by the simulator are indistinguishable. This naturally implies some kind of hardness, and one can ask whether or not this hardness suffices to construct concepts that are hard to learn not in the simulation sense but in the PAC model.

In this chapter we address these questions and show that in some sense non-trivial zero knowledge proofs are inherently "harder" than hard PAC learning problems in the standard model. On the other hand, it turns out that hardness of learning problems in the circuit model (*i.e.* CircCons, see Definition 3.4.1) are "as hard" as non-trivial zero knowledge proofs.

This mirrors the situation we saw in the previous chapter: AIOWF are "harder" than the standard notion of learning, but the circuit notion of learning is just as

hard as AIOWF. Indeed, to connect the learning and zero knowledge, we will use the Ostrovsky-Wigderson theorem Theorem 2.4.4 and the theorems proven in Chapter 3 regarding the relationship between learning and AIOWF.

This chapter is structured as follows. In Section 4.1 we observe that **ZK** $\neq$ **BPP** implies that PAC learning is hard using Theorem 3.2.1. In Section 4.2.1 we prove that basing **ZK** $\neq$ **BPP** on the hardness of learning is unlikely via relativizing proofs. In Section 4.2.2 and Section 4.2.3 we extend this to show that black-box GMW-style constructions of zero-knowledge proofs for **NP** based on hardness of learning are unlikely to exist. In contrast, in Section 4.3 we show that if $\mathsf{CircCons}_{\alpha,\beta} \in$ **SZKA** for appropriate values of α, β, and therefore if $\mathsf{CircCons}_{\alpha,\beta}$ is hard then **ZK** $\neq$ **BPP**.

4.1 ZK $\neq$ BPP implies hardness of learning

Using the Ostrovsky-Wigderson theorem Theorem 2.4.4, combined with Theorem 3.2.1, it follows easily that

Corollary 4.1.1. *If* **ZK** $\neq$ **BPP**, *then testing PAC* $(\frac{1-\varepsilon}{2})$*-consistency is hard against uniform (resp. non-uniform) algorithms, where* $\varepsilon = 2^{-n/4}$.

In particular, this means **ZK** $\neq$ **BPP** implies that PAC learning is hard. In the next section, we ask the reverse question: does hardness in the PAC model suffice to build zero knowledge proofs for non-trivial languages (*i.e.* languages outside **BPP**)?

4.2 Can ZK $\neq$ BPP be based on hardness of learning?

4.2.1 Relativizing techniques

We already saw in Theorem 3.3.1 that there is an oracle relative to which testing PAC learnability is hard and yet AIOWF do not exist. Combining Corollary 3.3.7 with the Ostrovsky-Wigerson theorem (Theorem 2.4.4) (whose proof is relativizing), we obtain our main theorem about relativizing proofs for zero knowledge.

Theorem 4.2.1. *There exists an oracle $\mathcal{O}$ relative to which testing PAC consistency is hard over the uniform distribution with membership queries, but* $\mathbf{ZK}^{\mathcal{O}} = \mathbf{BPP}^{\mathcal{O}}$.

The Ostrovsky-Wigderson theorem

As mentioned earlier, the Ostrovsky-Wigderson theorem (Theorem 2.4.4) is relativizing and efficient-oracle, but not fully black-box. We sketch the proof in order to point out the precise argument that is non-black-box: supposing that there exist no AIOWF, we show that $\mathbf{ZK} = \mathbf{BPP}$. Fix any $L \in \mathbf{ZK}$ with simulator S. It suffices to show that the "simulation-based prover" is efficiently computable: the simulation-based prover is defined by the conditional distribution of the simulator. Given a prefix of messages $m_1, \ldots, m_i$ (say m_i is a verifier message), the simulated prover samples a message m_{i+1} according to the distribution $S(x, U_r)$ conditioned on the first i messages being $m_1, \ldots, m_i$. If one could efficiently compute the simulated prover distribution (or approximate it) then this would give an algorithm for L: run the honest verifier and interact it with the simulated prover. By the zero-knowledge property the verifier will accept $x \in L$, and by soundness the verifier will reject $x \notin L$.

We show how to approximate the simulated prover assuming AIOWF do not exist. Let $S_i(x, \omega)$ be the function that takes random coins ω outputs the first i messages of the simulator. Suppose that the i'th message is sent by the receiver, then one way to sample the simulated prover's $i+1$'th message in response to a partial transcript $\tau_i = (m_1, \ldots, m_i)$ is to first invert $S_i(x, \cdot)$ on τ_i to obtain random coins ω such that $S_i(x, \omega) = \tau_i$, and then compute $S_{i+1}(x, r)$ and output the $i+1$'th message.

Assuming that AIOWF do not exist and using the equivalence with AIDOWF (Remark 2.3.9), there is an efficient distributional inverter I_i such that the following two distributions are computationally indistinguishable:

$$D_{\mathsf{Sim}} = (\tau_i, I_i(\tau_i)) \text{ and } D_{\mathsf{Honest}} = (\tau_i', I_i(\tau_i')) \tag{4.2.1}$$

Here, τ_i is generated using the simulated prover and τ_i' is generated using the honest prover. The fact that the inversion procedure is *efficient* is critical because we only have the guarantee that the output of the simulator is *computationally* indistinguishable from the honest transcript. If I_i were inefficient, then D_{Sim} and D_{Honest} may be distinguishable since it is conceivable that the honest transcript and the simulator transcript are computationally indistinguishable but have disjoint support, in which case inverting an honest transcript as if it were output by the simulator is information-theoretically impossible. Inductively using this observation for $\tau_1, \ldots, \tau_c$ where c is the number of rounds in the protocol establishes the theorem, since at the end the honest transcript is computationally indistinguishable from the transcript obtained by interacting the verifier with the simulated prover.

It is clear this proof is not black-box since, as noted above, the proof uses the fact that the inverter is efficient in a critical way. However, it is efficient-oracle black-box since the assumption that the inverter is efficient is the *only* way in which we "use the code" of the oracle. Furthermore, it is relativizing: if the prover, verifier, simulator,

and distinguisher are given access to an oracle and the AIOWF inverter is also allowed access to the oracle, the same reasoning goes through.

4.2.2 Black-box ZK proofs based on hardness of learning

Unfortunately, in this setting ruling out relativizing proofs is not very convincing because we have non-relativizing proofs that base $\mathbf{ZK} \neq \mathbf{BPP}$ on various complexity assumptions. In particular the celebrated result of Goldreich, Micali, and Wigderson [GMW86], which proves that **NP** has a zero knowledge protocol based on the existence of one-way functions, does not relativize because they work directly with the explicit **NP**-complete problem Three Coloring (3-COL).

Black-box proofs: [GMW86] does not relativize, but it is black-box: they require only black-box access to a one-way function to construct a zero-knowledge protocol for 3-COL. Our next result rules out *black-box proofs* that zero knowledge is non-trivial based on the hardness of learning. Our result applies to construction-black-box proofs as well as fully-black-box proofs, although the proof for the construction-black-box case is more involved.

Unlike for the case of AIOWF (Theorem 3.3.1), our results do not show that GMW-style black-box proofs are impossible because there *are* zero knowledge protocols whose security is unconditional (*e.g.* for Graph Isomorphism, Quadratic Residuosity). It is conceivable that even 3-COL has such a protocol (*i.e.* $\mathbf{NP} \subseteq \mathbf{SZK}$), in which case its security proof would use no complexity assumptions and hence would be trivially black-box. This is considered unlikely since by the fact that $\mathbf{SZK} \subseteq \mathbf{AM} \cap \mathbf{coAM}$ (Theorem 2.4.3) this would imply that the polynomial hierarchy collapses and contradict Conjecture 1.2.1. We prove that this is the only possibility:

Theorem 4.2.2. *If there exists a construction-black-box proof that constructs a* **ZK**

protocol for a language L assuming PAC learning is hard, then in fact $L \in \mathbf{SZK}$.

Under the standard conjecture that $\mathbf{NP} \not\subseteq \mathbf{SZK}$ (a consequence of Conjecture 1.2.1), Theorem 4.2.2 says that such proofs for an **NP**-complete language L cannot exist.

We first prove the result for fully-black-box reductions, then explain how to extend the proof to construction-black-box reductions.

Proof of fully-black-box case. A fully-black-box proof is relativizing, so both the construction and analysis must hold relative to any oracle. We will use the same oracle from Definition 3.3.3. We recall the definition here:

Definition 3.3.3 (Restated)**.** $\mathcal{O}$ is drawn from the following the distribution. First, for each n select a function $\mathcal{R}^{(n)} : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ by letting each $z \in \{0,1\}^n$ be a *hard instance* with probability $2^{-n/2}$, where we set $\mathcal{R}_z = \mathcal{R}^{(n)}(z, \cdot)$ to be a random function, and letting z be an *easy instance* with probability $1 - 2^{-n/2}$, where $\mathcal{R}_z \equiv 0$. Let $\mathcal{O}$ decide $\mathsf{QBF}_*^{\mathcal{R}}$, which is $\mathbf{PSPACE}_*^{\mathcal{R}}$-complete.

Recall that Lemma 3.3.5 says with probability 1 over the choice of $\mathcal{R}$, $F = \{\mathcal{R}_z\}_{z \in \{0,1\}^*}$ is hard to learn. By our hypothesis, this implies $L \in \mathbf{ZK}^{\mathcal{O}}$, and furthermore in the zero-knowledge protocol for L, the prover, verifier, and simulator all use access only to the hard concept class F, which can be implemented using just $\mathcal{R}$ (and not $\mathcal{O}$) gates.

Next, we claim that not only is the protocol computationally zero knowledge, it is statistically zero knowledge. Formally, applying the relativized version of the SZK/AIOWF characterization (Theorem 2.4.7) we know that if $L \in \mathbf{ZK}^{\mathcal{O}}$ then (a) there is an efficient reduction Red reducing L to $\mathsf{SD}^{\mathcal{O}}$, or (b) there exists AIOWF against non-uniform inverters relative to $\mathcal{O}$. Case (b) never occurs because Lemma 3.3.6 tells us that AIOWF do not exist relative to $\mathcal{O}$, so we must be in case (a).

In fact, the proof of Theorem 2.4.7 actually proves not only that Red reduces L to $\mathsf{SD}^{\mathcal{O}}$ but the circuits that Red produce are defined simply in terms of the (code of the) simulator of the original $\mathbf{ZK}^{\mathcal{O}}$ protocol. But the simulator of the original protocol needed access only to $\mathcal{R}$. Therefore, we actually can conclude that with probability 1 over the choice of $\mathcal{R}$, Red reduces every $x \in L$ to a YES instance of $\mathsf{SD}^{\mathcal{R}}$ and every $x \notin L$ to a NO instance of $\mathsf{SD}^{\mathcal{R}}$. (Observe that reducing to $\mathsf{SD}^{\mathcal{R}}$ is crucial: reducing L to $\mathsf{SD}^{\mathcal{O}}$ instead would be meaningless, since $\mathsf{SD}^{\mathcal{O}}$ is complete for $\mathbf{SZK}^{\mathcal{O}}$. $\mathbf{SZK}^{\mathcal{O}}$ contains $\mathbf{PSPACE}$ since $\mathcal{O}$ can decide QBF, so proving $L \subseteq \mathbf{SZK}^{\mathcal{O}}$ does not lead to any implausible conclusions. In fact, this is why one cannot use say Theorem 2.4.4 to conclude that the non-existence of AIOWF implies that $L \in \mathbf{BPP}$; applying it would only show $L \in \mathbf{BPP}^{\mathcal{O}}$, which is meaningless since $\mathbf{PSPACE} \subseteq \mathbf{BPP}^{\mathcal{O}}$.)

We can now deduce that with high probability over $\mathcal{R}$, the reduction Red is good for all long enough instances. Let us say that "Red succeeds on L_n" if for all x of length n, $\mathsf{Red}(x)$ maps each $x \in L$ to a YES instance of $\mathsf{SD}^{\mathcal{R}}$ and each $x \notin L$ reduction to a NO instance of $\mathsf{SD}^{\mathcal{R}}$ (*i.e.* they satisfy the promise of $\mathsf{SD}^{\mathcal{R}}$).

Claim 4.2.3. *If* Red *reduces* L *to* $\mathsf{SD}^{\mathcal{R}}$ *with probability* 1 *over* $\mathcal{R}$*, then*

$$\Pr_{\mathcal{R}}[\mathsf{Red} \text{ succeeds on } L_n] \to 1 \text{ as } n \to \infty$$

To prove the claim, let A_n be the event that Red succeeds on $L_{\geq n}$, *i.e.* Red succeeds on all inputs of length at least n (rather than exactly n). Notice that it suffices to show $\Pr[A_n] \to 1$ as $n \to \infty$. We know by hypothesis that $1 = \Pr_{\mathcal{R}}[\mathsf{Red} \text{ reduces } L \text{ to } \mathsf{SD}^{\mathcal{R}}] \leq \Pr_{\mathcal{R}}[\bigcup_{i=1}^{\infty} A_i]$. Since $A_n \subseteq A_{n+1}$, we have that:

$$\Pr\left[\bigcup_{i=1}^{\infty} A_i\right] = \sum_{i=1}^{\infty} \Pr[A_i \wedge \overline{A_{i-1}}]$$

But since $\Pr[A_n] = \sum_{i=1}^{n} \Pr[A_i \wedge \overline{A_{i-1}}]$, the claim follows.

Lemma 4.2.4. *If for sufficiently large n, $\Pr_{\mathcal{R}}[\mathsf{Red}$ succeeds on $L_n] > 99/100$, then $L \in$* **SZK**.

Claim 4.2.3 means that the hypothesis of this lemma is satisfied, and so the lemma implies fully-black-box case of Theorem 4.2.2. ■

Proof of Lemma 4.2.4. We have by hypothesis that Red efficiently maps each input x to an instance of $\mathsf{SD}^{\mathcal{R}}_{\alpha,\beta}$, say with $\alpha = 99/100$ and $\beta = 1/100$. By padding, we can assume without loss of generality that the input and output length of $X_i^{\mathcal{R}}$ is n, and $|X_i| \leq p(n) = \mathrm{poly}(n)$ for $i = 0, 1$.

By hypothesis, with probability $99/100$ over the choice of $\mathcal{R}$, for every x of length n, $x \in L$ reduces to $(X_0^{\mathcal{R}}, X_1^{\mathcal{R}})$ such that $\Delta(X_0^{\mathcal{R}}, X_1^{\mathcal{R}}) > 99/100$ while $x \notin L$ reduces to $(X_0^{\mathcal{R}}, X_1^{\mathcal{R}})$ such that $\Delta(X_0^{\mathcal{R}}, X_1^{\mathcal{R}}) < 1/100$.

Claim 4.2.5. *There is an efficient deterministic reduction Red' such that for all $x \in L$, $\mathsf{Red}'(x) = (X_0', X_1')$ satisfies $\Delta(X_0', X_1') > 24/25$ and for all $x \notin L$, $\mathsf{Red}'(x) = (X_0', X_1')$ satisfies $\Delta(X_0', X_1') < 1/25$.*

Since $(24/25)^2 > 1/25$, this is still in **SZK** and so the claim shows that Red' puts $L \in$ **SZK**.

To prove the claim, let Red' work by first running Red to produce $(X_0^{\mathcal{R}}, X_1^{\mathcal{R}})$, and then transforming those circuits the following way. Let Q be a circuit that takes some random bits and generates a "fake" oracle $\mathcal{R}_Q$ whose distribution on inputs of up to length $2 \log 10^8 p(n)$ is identical to the real distribution $\mathcal{R}$, and for inputs longer than $2 \log 10^8 p$ always returns 0. It is clear $\mathcal{R}_Q$ can be described and evaluated in polynomial time, and there is a circuit Q of size $\mathrm{poly}(p)$ that constructs $\mathcal{R}_Q$ using $m = \mathrm{poly}(p)$ random bits.

$\mathsf{Red}'(x) = (X_0', X_1')$ where X_0' is the circuit that takes $m + n$ random bits and uses m

bits (call these m bits ω) for Q to generate a fake random oracle $\mathcal{R}_Q$, and uses n bits to sample a random $x \leftarrow_{\mathrm{R}} X_0^{\mathcal{R}_Q}$ and then outputs (ω, x). X_1' is the circuit that takes $m+n$ random bits just as above except it outputs (ω, x) where $x \leftarrow_{\mathrm{R}} X_1^{\mathcal{R}_Q}$.

We prove that Red' satisfies the claim. Let X be either X_0 or X_1 (the same argument applies to both). For $r \in \{0,1\}^n$, let $B(r)$ be the bad event over the choice of $\mathcal{R}$ that $X^{\mathcal{R}}(r)$ queries a hard instance z of length $> 2\log 10^8 p$, and $B_i(r)$ be the event that the i'th oracle query of $X^{\mathcal{R}}(r)$ (for some arbitrary ordering of the queries) is a hard instance z of length $> 2\log 10^8 p$. Recall from the definition of $\mathcal{R}$ that z of length ℓ are hard with probability $2^{-\ell/2}$. It holds that:

$$\Pr_{\mathcal{R}, r \leftarrow_{\mathrm{R}} U_n}[B(r)] = \mathbb{E}_r \Pr_{\mathcal{R}}[B(r)] \leq \mathbb{E}_r \sum_{i=1}^{p} \Pr_{\mathcal{R}}[B_i(r)] \leq 1/10^8$$

since over the randomness of $\mathcal{R}$, the probability that any query of length $> 2\log 10^8 p$ is hard is at most $1/(10^8 p)$.

Now by Markov, we have that

$$\Pr_{\mathcal{R}}[\Pr_{r \leftarrow_{\mathrm{R}} U_n}[B(r)] > 1/10^4] < 1/10^4$$

Notice that for good $\mathcal{R}$ where $B(r)$ occurs with probability $\leq 1/10^4$, we have that $\Delta(X^{\mathcal{R}}, X^{\mathcal{R}_Q}) \leq 1/10^4$. Therefore, with probability $> 99/100 - 2/10^4$ we get a good fixing of ω used by Q to generate $\mathcal{R}_Q$, where by good we mean that

$$x \in L \Rightarrow \Delta(X_0^{\mathcal{R}_Q}, X_1^{\mathcal{R}_Q}) > 99/100 - 2/10^4$$
$$x \notin L \Rightarrow \Delta(X_0^{\mathcal{R}_Q}, X_1^{\mathcal{R}_Q}) < 1/100 + 2/10^4$$

Therefore, the claim follows by averaging over all ω and using the fact that a $\frac{99}{100} - \frac{2}{10^4}$ fraction of the ω are good, so that

$$x \in L \Rightarrow \Delta(X_0', X_1') > (\tfrac{99}{100} - \tfrac{2}{10^4})(\tfrac{99}{100} - \tfrac{2}{10^4}) > \tfrac{24}{25}$$
$$x \notin L \Rightarrow \Delta(X_0', X_1') < \tfrac{1}{100} + \tfrac{2}{10^4} + \tfrac{1}{100} + \tfrac{2}{10^4} < \tfrac{1}{25}$$

This concludes the proof of the lemma. ■

4.2.3 Construction-black-box ZK proofs

The proof above fails to rule out construction-black-box reductions because we use Lemma 3.3.6, which says any efficiently computable function can be inverted by an adversary with access to $\mathcal{O}$. In contrast, in a construction-black-box reduction the adversary is allowed access *only to the hard concept class*, which in the above proof is $F = \{\mathcal{R}_z\}$. To rule out construction-black-box reductions we will "embed" $\mathbf{PSPACE}^{\mathcal{R}}_*$ inside the hard concept class itself (an idea of Simon [Sim98], see also [RTV04]), but this must be done carefully.

We cannot simply use the observation used in Theorem 3.3.14 that the set of all $\mathsf{QBF}^{\mathcal{R}}_*$ formulas are hard-to-learn given access to an oracle $\mathcal{O}$ that decides $\mathsf{QBF}^{\mathcal{R}}_*$. With such an oracle, the inverter for the AIOWF is indeed able to invert all AIOWF, but the verifier in the zero knowledge protocol is also able to access $\mathcal{O}$, which means that the verifier can trivially decide **PSPACE**-complete problems on its own without the prover's help. From this we can only conclude that $L \in \mathbf{PSPACE}$, which is uninteresting.

To achieve a meaningful result, we must carefully balance the power of the oracle to satisfy three requirements:

1. The oracle constitutes a collection of hard-to-learn functions.
2. Using the oracle, it is possible to invert all circuits, thus breaking all AIOWF.
3. The verifier in the zero knowledge proof is unable to use the oracle to decide hard languages.

The key to achieve these two conflicting goals simultaneously is that the SZK/AIOWF characterization (Theorem 2.4.7) allows the AIOWF inverter to be *non-uniform*, while the verifier in the zero knowledge proof is uniform. Our oracle will be as follows:

Definition 4.2.6. Let $\mathcal{R} : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ be chosen as in Definition 3.3.3. Pick a random $z_0 \in \{0,1\}^n$, we call such z_0 "powerful" instances. Let $\mathcal{O} : \{0,1\}^{n+1} \times \{0,1\}^n \to \{0,1\}$ be defined as:

$$\mathcal{O}(z,b,x) = \begin{cases} \mathcal{R}_z(x) & b = 0 \\ \mathsf{QBF}_*^{\mathcal{R}}(\varphi) & b = 1, z = z_0, x = 0^{n-\sqrt{n}}\varphi \\ 0 & \text{else} \end{cases}$$

where φ is interpreted as a $\mathsf{QBF}_*^{\mathcal{R}}$ formula.

Testing PAC consistency is hard relative to $\mathcal{O}$

As with the oracle of Definition 3.3.3, we will prove that relative to the oracle above there exists a hard-to-learn concept class but AIOWF do not exist. Let $\mathcal{O}_{z,b} : \{0,1\}^n \to \{0,1\}$ be the function $\mathcal{O}(z,b,\cdot)$. The new hard-to-learn concept class will be $F = \{\mathcal{O}_{z,b}\}$; notice F contains a $\mathsf{QBF}_*^{\mathcal{R}}$ oracle (namely the functions $\mathcal{O}_{z_0,1}$), and it is insufficient to take $\{\mathcal{R}_z\}$ since it does not contain a $\mathsf{QBF}_*^{\mathcal{R}}$ oracle.

Lemma 4.2.7. *For any $\varepsilon \geq 2^{-\log^2 n}$, with probability 1 over $\mathcal{O}$, testing PAC $(\frac{1-\varepsilon}{2})$-consistency with a membership oracle is hard for the class $F = \{\mathcal{O}_{z,b}\}$ relative to $\mathcal{O}$.*

Proof. Any efficient circuit that can correctly test PAC consistency of $F = \{\mathcal{O}_{z,b}\}$ must necessarily correctly test PAC consistency of $\{\mathcal{O}_{z,0}\}$. Furthermore, $\mathcal{O}_{z,0} = \mathcal{R}_z$, so it suffices to bound the probability that $C^{\mathcal{O}}$ correctly tests PAC $(\frac{1-\varepsilon}{2})$-consistency of $\{\mathcal{R}_z\}$.

Abusing notation slightly, let $\mathsf{QBF}_*^{\mathcal{R}}$ denote an oracle deciding the language $\mathsf{QBF}_*^{\mathcal{R}}$. We observe that given any circuit C with $\mathcal{O}$ gates, we can emulate C with a circuit C' with $\mathsf{QBF}_*^{\mathcal{R}}$ gates. Suppose $|C| \leq s(n)$, then C' will be of size $\text{poly}(s(n))$. C' will include a hard-wiring of all the powerful instances z_0 up to length $s(n)$. C' emulates C in a straightforward way: whenever C queries $\mathcal{O}(z, 0, x)$, C' queries its $\mathsf{QBF}_*^{\mathcal{R}}$ gate with $\mathcal{R}_z(x)$, and whenever C queries $\mathcal{O}(z, 1, 0^{|z|-\sqrt{|z|}}\varphi)$, C' checks if z is the powerful instance of length $|z|$ and if so it queries $\mathsf{QBF}_*^{\mathcal{R}}(\varphi)$.

Therefore, if $C^{\mathcal{O}}$ is a polynomial size circuit that tests PAC consistency of F, so is $(C')^{\mathsf{QBF}_*^{\mathcal{R}}}$. But in Lemma 3.3.5 we already bounded the probability that such $(C')^{\mathsf{QBF}_*^{\mathcal{R}}}$ could test PAC consistency of $\{\mathcal{R}_z\}$ by $2^{-2^{\Omega(n)}}$! Therefore the probability that $C^{\mathcal{O}}$ can test PAC consistency is also bounded by $2^{-2^{\Omega(n)}}$.

■

Inverting AIOWF is easy

Second, we show that AIOWF do not exist relative to $\mathcal{O}$.

Lemma 4.2.8. *With probability* 1 *over* $\mathcal{O}$*, there exist no AIOWF against non-uniform adversaries relative to* $\mathcal{O}$*.*

Proof. Suppose f is a circuit with $\mathcal{O}$ gates of size at most $s(n)$. Then clearly f can be emulated by a circuit f' with $\mathsf{QBF}_*^{\mathcal{R}}$ gates of size $\text{poly}(s(n))$ that has hardwired all the powerful instances z_0 of length up to $s(n)$ and f' emulates f in the same way as C' emulates C in the proof of Lemma 4.2.7.

But now f' is just a function with $\mathsf{QBF}_*^{\mathcal{R}}$ gates, and therefore Lemma 3.3.6 tells us that there is an efficient inverter I that uses a $\mathsf{QBF}_*^{\mathcal{R}}$ oracle and inverts f'. But we have only a $\mathcal{O}$ oracle, not a $\mathsf{QBF}_*^{\mathcal{R}}$ oracle. To simulate a $\mathsf{QBF}_*^{\mathcal{R}}$ oracle, let $p(n)$

be an upper bound on the size of queries that I makes to $\mathsf{QBF}_*^{\mathcal{R}}$ given an input of length n. Let z_0 be a powerful instance of length $O(p(n)^2)$, and let I_{z_0} denote a circuit that emulates I except that whenever I queries φ to $\mathsf{QBF}_*^{\mathcal{R}}$, I_{z_0} sends the query to $\mathcal{O}(z_0, 1, 0^{p(n)^2-p(n)}\varphi)$. Since $\mathcal{O}(z_0, 1, 0^{p(n)^2-p(n)}\varphi) = \mathsf{QBF}_*^{\mathcal{R}}(\varphi)$, it follows that I_{z_0} successfully inverts if and only if I does, and therefore the lemma follows from Lemma 3.3.6. ■

Ruling out construction-black-box GMW-style constructions

Proof of Theorem 4.2.2, construction-black-box case. Let $\mathcal{O}$ be drawn from the distribution given by Definition 4.2.6. Lemma 4.2.7 and the hypothetical construction-black-box reduction implies that $L \in \mathbf{ZK}^{\mathcal{O}}$.

Using Theorem 2.4.7 and the fact that AIOWF do not exist (Lemma 4.2.8), we deduce that there is an efficient reduction Red such that with probability 1 over $\mathcal{O}$, Red reduces L to $\mathsf{SD}^{\mathcal{O}}$. As before, we claim that if Red reduces L to $\mathsf{SD}^{\mathcal{O}}$ with probability 1 over $\mathcal{O}$, then $\Pr_{\mathcal{O}}[\mathsf{Red}$ succeeds on $L_n] \to 1$ as $n \to \infty$. The proof is identical to the proof of Claim 4.2.3.

Since for large enough n, Red succeeds on L_n with probability 99/100 over the choice of $\mathcal{O}$, and we can then hardwire $\mathcal{O}$ to place $L \in \mathbf{SZK}$:

Claim 4.2.9. *If for all large enough n the reduction Red succeeds on L_n with probability 99/100 over the choice of $\mathcal{O}$, then $L \in \mathbf{SZK}$.*

Proof. The claim is proven by reducing L to SD using the same argument as in the proof of Lemma 4.2.4 modulo two differences: first, the bad events $B_i(r)$ are redefined to be the event that the i'th unique z that $X^{\mathcal{O}}(r)$ queries is either hard or powerful, which means that we get a slightly worse bound that $\Pr[B(r)] \leq 1/10^8 + 1/10^{16}$. The

rest of the calculations lose a small but unimportant factor because of this.

Second, in order for the circuit Q to sample the fake oracle $\mathcal{O}_Q$ identically distributed to $\mathcal{O}$ on all inputs up to length $2\log 10^8 p = O(\log n)$ and 0 on longer inputs, Q must be able to decide $\mathsf{QBF}^{\mathcal{R}}_*$ formulas of size up to $\sqrt{2\log 10^8 p} = O(\sqrt{\log n})$. It suffices to note that Q can do this in polynomial size simply by brute force, because $\mathsf{QBF}^{\mathcal{R}}_*$ can be decided in time $2^{O(n^2)}$ (using say Proposition A.1.1) and the inputs here are of size $O(\sqrt{\log n})$. ■

This concludes the proof of Theorem 4.2.2. ■

4.3 CircCons $\in$ ZK

Already Corollary 3.5.2 and Proposition 3.4.4 imply the following

Corollary 4.3.1. *If* CircCons *is hard against uniform (resp. non-uniform) algorithms, then there exist AIOWF against uniform (resp. non-uniform) inverters.*

We will show an even tighter relationship: $\mathsf{CircCons}_{\alpha,\beta} \in \mathbf{SZKA}$ for any α, β where $(1-2\alpha)^2 > 1-2\beta$.

4.3.1 StatCircCons

We first show that a statistical version of CircCons, which we call StatCircCons, is actually in **SZKP**, and then we use the characterization of **SZKA** given in Theorem 2.4.7 to extend the result to CircCons. The difference between StatCircCons and CircCons is that for StatCircCons we only care whether or not a good deterministic labeling for (X, Y) exists; we do not care whether that labeling is efficient.

Definition 4.3.2. $\mathsf{StatCircCons}_{\alpha,\beta}$ is the promise problem with input a circuit C sampling a joint distribution (X, Y) such that:

- YES instance: $\mathsf{err}((X, Y), \mathcal{F}_{\mathsf{all}}) < \alpha$
- NO instance: $\mathsf{err}((X, Y), \mathcal{F}_{\mathsf{all}}) > \beta$

Lemma 4.3.3. *There is an efficient Karp reduction from* $\mathsf{StatCircCons}_{\alpha,\beta}$ *to* $\mathsf{SD}_{\alpha',\beta'}$ *where* $\alpha' = (1 - 2\alpha)(1 - 2^{-n})$ *and* $\beta' = (1 - 2\beta)(1 - 2^{-n})$.

Proof. Let C be an instance of $\mathsf{StatCircCons}_{\alpha,\beta}$ and let (X, Y) be the distribution that C samples. We can assume without loss of generality that the marginal distribution of Y is unbiased; otherwise, modify C to output the distribution $(X \circ b, Y \oplus b)$ where b is an unbiased coin flip. It is easy to check that $\mathsf{err}((X \circ b, Y \oplus b), \mathcal{F}_{\mathsf{all}}) = \mathsf{err}((X, Y), \mathcal{F}_{\mathsf{all}})$. We construct an instance (X_0, X_1) of SD from (X, Y) as follows. X_b samples n times from (X, Y) to obtain $(x_1, y_1), \ldots, (x_n, y_n)$ and outputs a random x_i such that $y_i = b$. If none of the $y_i = b$, then it outputs a special failure symbol $\perp$. We claim that

$$\Delta(X_0, X_1) = (1 - 2^{-n})(1 - 2\mathsf{err}((X, Y), \mathcal{F}_{\mathsf{all}})) \tag{4.3.1}$$

To prove this, first observe that

$$\Delta(X_0, X_1) = (1 - 2^{-n})\Delta((X \mid Y = 0), (X \mid Y = 1)) \tag{4.3.2}$$

since conditioned on not outputting $\perp$, it holds that X_b is distributed identically to $X \mid Y = b$. Next, by the definition of statistical distance, we have that

$$\begin{aligned}\Delta((X \mid Y = 0), (X \mid Y = 1)) &= \max_T \{\Pr[T(X) = 1 \mid Y = 1] - \Pr[T(X) = 1 \mid Y = 0]\} \\ &= \max_T \{1 - \Pr[T(X) = 0 \mid Y = 1] - \Pr[T(X) = 1 \mid Y = 0]\}\end{aligned}$$

Observe that because Y is balanced, it holds that

$$\Pr[T(X) = 0 \mid Y = 1] + \Pr[T(X) = 1 \mid Y = 0] = 2\Pr[T(X) \neq Y]$$

Therefore we may write

$$\Delta((X \mid Y = 0), (X \mid Y = 1)) = \max_T \{1 - 2\Pr[T(X) \neq Y]\} \tag{4.3.3}$$

$$= 1 - 2\min_T \Pr[T(X) \neq Y] \tag{4.3.4}$$

$$= 1 - 2\ \mathsf{err}((X, Y), \mathcal{F}_{\mathsf{all}}) \tag{4.3.5}$$

Combining Equation 4.3.2 with Equation 4.3.5 we obtain Equation 4.3.1.

■

In fact, one can show the converse as well:

Lemma 4.3.4. $\mathsf{SD}_{\alpha,\beta}$ *efficiently reduces to* $\mathsf{StatCircCons}_{\frac{1-\alpha}{2},\frac{1-\beta}{2}}$.

Proof. We can run the proof of Lemma 4.3.3 "in reverse". Given an instance (X_0, X_1) of $\mathsf{SD}_{\alpha,\beta}$, construct the circuit C that samples from the following distribution: sample $b \leftarrow_{\mathrm{R}} \{0,1\}$, then sample a random $x \leftarrow_{\mathrm{R}} X_b$, and output (x, b). Let this distribution be denoted (X, Y). Then we can write:

$$\begin{aligned}
\mathsf{err}((X, Y), \mathcal{F}_{\mathsf{all}}) &= \min_T \Pr[T(X) \neq Y] \\
&= \min_T \left\{ \frac{\Pr[T(X) = 0 \mid Y = 1] + \Pr[T(X) = 1 \mid Y = 0]}{2} \right\} \\
&= \min_T \left\{ \frac{1 - \Pr[T(X) = 1 \mid Y = 1] + \Pr[T(X) = 1 \mid Y = 0]}{2} \right\} \\
&= \frac{1 - \max_T \{\Pr[T(X) = 1 \mid Y = 1] - \Pr[T(X) = 1 \mid Y = 0]\}}{2} \\
&= \frac{1 - \Delta(X_0, X_1)}{2}
\end{aligned}$$

■

As an immediate corollary, we have the following.

Theorem 4.3.5. $\mathsf{StatCircCons}_{\alpha,\beta}$ *is* **SZKP***-complete for any constants* α, β *such that* $(1-2\alpha)^2 > (1-2\beta)$.

Proof. By Lemma 4.3.3 we can reduce $\mathsf{StatCircCons}_{\alpha,\beta}$ to $\mathsf{SD}_{\alpha',\beta'}$. Since $\alpha' = (1-2^{-n})(1-2\alpha), \beta' = (1-2^{-n})(1-2\beta)$ and α, β are constants, this means that for large enough n, it holds that $(\alpha')^2 > \beta'$, and therefore $\mathsf{StatCircCons}_{\alpha,\beta} \in$ **SZKP**. On the other hand, we know that $\mathsf{SD}_{1-2\alpha,1-2\beta}$ for $(1-2\alpha)^2 > 1-2\beta$ is complete for **SZK** and it reduces to $\mathsf{StatCircCons}_{\alpha,\beta}$, therefore $\mathsf{StatCircCons}_{\alpha,\beta}$ is also **SZK**-hard. ■

Corollary 4.3.6. $\mathsf{StatCircCons}_{\alpha,\beta} \in$ **coAM** *for any* α, β *such that there exists a polynomial* $p(n)$ *such that* $\beta - \alpha > 1/p(n)$.

Proof. First apply Lemma 4.3.3 which reduces $\mathsf{StatCircCons}_{\alpha,\beta}$ to $\mathsf{SD}_{\alpha',\beta'}$ for $\alpha' = (1-2^{-n})(1-2\alpha)$ and $\beta' = (1-2^{-n})(1-2\beta)$, then apply the fact that for $\alpha' - \beta' > 1/p(n)$ it holds that $\mathsf{SD}_{\alpha',\beta'} \in$ **coAM**. For completeness we include a proof of this fact in Lemma A.4.1. ■

4.3.2 CircCons

We now use Lemma 4.3.3 to show that $\mathsf{CircCons}_{\alpha,\beta} \in$ **SZKA** for α, β satisfying $(1-2\alpha)^2 > (1-2\beta)$. The idea is to use the SZK/AIOWF characterization of Theorem 2.4.7, which says that a problem is in **SZKA** if and only if it can be decomposed into a part that is in **SZKP** and a part from which we can build AIOWF. We will decompose $\mathsf{CircCons}_{\alpha,\beta}$ into a part that is in $\mathsf{StatCircCons}_{\alpha,\beta}$ and a part that cannot be labeled correctly by polynomial-size circuits but can be labeled by an inefficient circuit, and we will show how to build AIOWF on this part.

Theorem 4.3.7. *For any efficiently computable class of functions F and any constants α, β satisfying $(1-2\alpha)^2 > (1-2\beta)$, it holds that* $\mathsf{CircCons}^F_{\alpha,\beta} \in \mathbf{SZKA}$.

Proof. Let $\gamma = (1-2\alpha)^2 - (1-2\beta)$. Let $\Pi = \mathsf{CircCons}^F_{\alpha,\beta}$ and let Π_Y be the YES instances and Π_N be the NO instances. We use the Theorem 2.4.7, which says that in order to prove $\mathsf{CircCons} \in \mathbf{SZKA}$ it suffices to prove that there exists a subset $W \subseteq \Pi_N$ and an efficiently computable function f mapping W to circuits such that

1. $\Pi' = (\Pi_Y, \Pi_N \setminus W) \in \mathbf{SZKP}$

2. For every efficient non-uniform algorithm A, it holds for all $C \in W$ that the circuit f_C, which takes m input bits, satisfies

$$\Pr_{x \leftarrow_R U_m}[A(f_C, y) \in f_C^{-1}(y) \mid y = f_C(x)] \leq n^{-\omega(1)}$$

We will show that the following set satisfies the above:

$$W = \{C \in \Pi_N \mid C \text{ samples } (X,Y) \; s.t. \; \mathsf{err}((X,Y), \mathcal{F}_{\mathsf{all}}) \leq \beta - \gamma/2\}$$

Proof of Item 1: $C \in \Pi_N \setminus W$ if and only if C samples a distribution (X,Y) such that $\mathsf{err}((X,Y), \mathcal{F}_{\mathsf{all}}) > \beta - \gamma/2$. Therefore $\Pi' \in \mathsf{StatCircCons}_{\alpha,\beta-\gamma/2}$ and since $(1-2\alpha)^2 > (1-2(\beta-\gamma/2))$, it holds by Lemma 4.3.3 that $\Pi' \in \mathbf{SZKP}$.

Proof of Item 2 for a circuit C sampling (X,Y), let C_1 be the subcircuit that samples X (*i.e.* the first n output bits of C). We claim that for all $C \in W$ and all efficient non-uniform inverters I, it holds that

$$\Delta((r, C_1(r)), (I(C_1, x), x \mid x = C_1(r))) > \Omega(\gamma^6/s)$$

where r is uniformly random. We prove this by contradiction: suppose not, then we could use Lemma 3.5.1 to build from I a hypothesis h of size $\mathrm{poly}(s/\gamma)$ such that

$\mathsf{err}((X,Y),h) \leq \mathsf{err}((X,Y),\mathcal{F}_{\mathsf{all}})+\gamma/2$. Since $C \in W$, this implies that $\mathsf{err}((X,Y),\mathcal{F}_{\mathsf{all}}) \leq \beta - \gamma/2$, and therefore $\mathsf{err}((X,Y),h) \leq \beta$. But h is of polynomial size $\mathrm{poly}(s/\gamma)$, so because C is a NO instance such h should not exist, a contradiction.

This implies that there exists AIDOWF, namely the family of functions computed by $\{C_1 \mid C \in W\}$ are hard to distributionally invert. By Remark 2.3.9 this means there exists AIOWF. ■

We remark that this proof of *non-relativizing* because we use the non-relativizing direction of Theorem 2.4.7. The non-relativizing nature stems from the use of the GMW protocol in Theorem 2.4.7.

Regarding the restriction $(1-2\alpha)^2 > (1-2\beta)$

We do not know whether our results that $\mathsf{StatCircCons}_{\alpha,\beta} \in \mathbf{SZKP}$ and $\mathsf{CircCons}_{\alpha,\beta} \in \mathbf{SZKA}$ generalize to the case where the condition $(1-2\alpha)^2 \leq (1-2\beta)$. The bottleneck is that it is unknown whether $\mathsf{SD}_{\alpha,\beta} \in \mathbf{SZKP}$ when $\alpha^2 \leq \beta$, and since we saw from Theorem 4.3.5 that $\mathsf{StatCircCons}_{\alpha,\beta}$ is basically equivalent to $\mathsf{SD}_{(1-2\alpha),(1-2\beta)}$, proving this fact for $\mathsf{StatCircCons}$ would resolve long-standing open problem about SD.

4.4 Summary

Figure 4.4 summarizes the relationship between zero knowledge, AIOWF, and learning.

As with the case of AIOWF, the main message here is that learning in the standard model is "harder" than zero knowledge, but learning in the $\mathsf{CircCons}$ or $\mathsf{CircLearn}$ model is not.

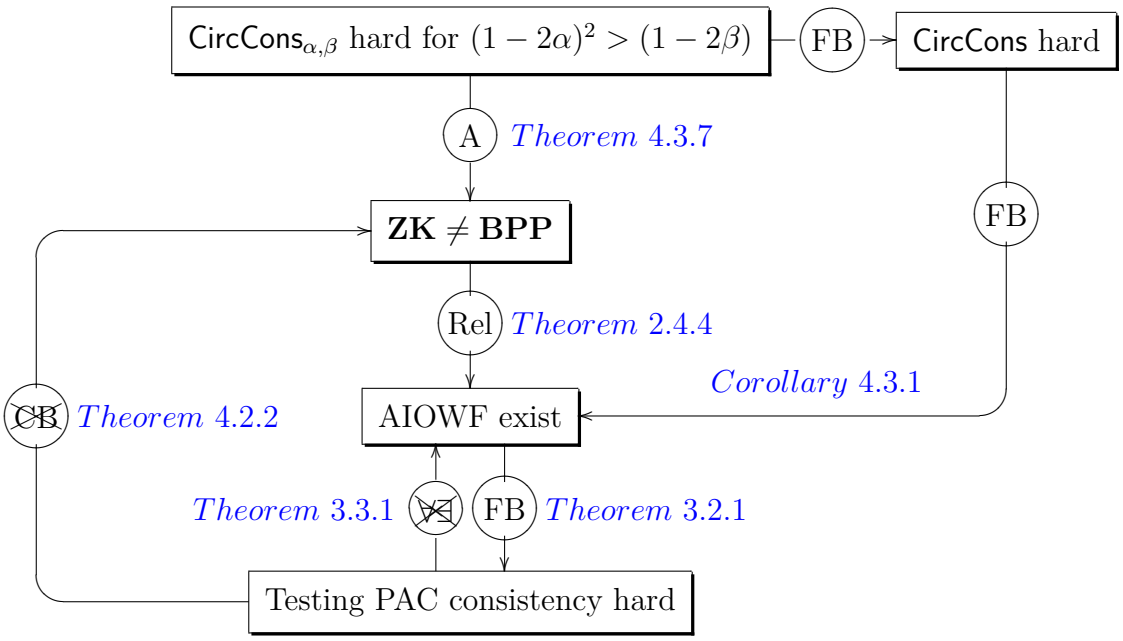

Figure 4.1: **ZK**, AIOWF and learning

Open questions: we saw in Theorem 4.3.5 that StatCircCons is complete for **SZK**. Can one show that CircCons characterizes **SZKA** in any meaningful way?

Chapter 5

Learning and NP

Computational learning theory has provided many strong algorithmic tools and showed that non-trivial concept classes are efficiently learnable. But despite these successes it seems that some (even simple) concept classes are hard to learn. It is considered all the more unlikely that *every* efficiently computable function can be learned efficiently. Ideally one could prove that PAC learning is hard, but this seems unattainable since it would imply that $\mathbf{P} \neq \mathbf{NP}$.

The next best thing would be to prove that PAC learning is **NP**-hard. In fact, one of the first questions Valiant asked after defining the PAC learning model was whether or not PAC learning is **NP**-hard, *i.e.* whether it is possible to prove that $\mathbf{P} \neq \mathbf{NP}$ implies that PAC learning is hard. Shortly afterwards, Pitt and Valiant [PV88] proved that indeed PAC learning the class of k-term DNF functions is **NP**-hard, but *only if the learning algorithm is required to be proper*, *i.e.* only if the learning algorithm is forced to output a k-term DNF. In fact, Valiant showed in his original paper [Val84] how to learn k-term DNF using k-CNF (CNF formulas where each clause has at most k literals) and therefore this problem is not hard in the general PAC model.

There have since been a long line of results showing that various problems are hard to learn if the output of the learning algorithm is proper or "almost proper" (where the hypothesis class is not polynomial-size circuits but is larger than the concept class) [BR92, HJLT96, BDEL03, ABF+04, Fel06a, Fel06b, GR06, GKS07]. One may hope to use these results as a starting point for proving **NP**-hardness in the general (*i.e.* improper) setting. Indeed, although some of the aforementioned hardness results seem useless for this purpose (as they apply to concept classes which are known to be improperly learnable), others might still be relevant. In particular, [ABF+04] show that it is **NP**-hard to learn the intersection of two halfspaces even in a semi-proper setting where the learner is allowed to use an intersection of any (constant) number of halfspaces. Similarly, [GKS07] show that learning parity in the agnostic model, where the data is noisy, is **NP**-hard even if the learner is allowed to use a low degree polynomial. These concept classes are not known to be efficiently learnable. Also, both works rely on highly non-trivial PCP machinery. This may give some hope that similar techniques will eventually prove that (improper) learning is **NP**-hard.

In this chapter we show that such hope is most likely unfounded, and that there are inherent limitations to proving that PAC learning (or even agnostic learning) is **NP**-hard when there is no restriction on the learning algorithm except that it be efficient. We will use the results of Chapter 3 to show that this question is closely related to the questions of basing cryptography on **NP**-hardness, which is a well-studied problem [FF93, BT06, AGGM06, Pas06]. Our results will focus on various kinds of reductions from **NP**-hardness to the hardness of learning, and will show that in many cases, standard techniques are insufficient to prove that PAC (or agnostic) learning is **NP**-hard.

First, we prove that if there exists a Karp reduction from **NP**-hardness to learn-

ing, then one can use an AIOWF inverter to decide **NP**-complete problems. As a corollary, Minicrypt collapses to Pessiland in Impagliazzo's hierarchy, or in other words the existence of a hard-on-average problem in **NP** implies that one-way functions exist. This suggests that such a reduction would have surprising consequences in cryptography. Under certain additional conditions, such a reduction also implies that $\mathbf{NP} \subseteq \mathbf{SZKA}$ or even that $\mathbf{NP} \subseteq \mathbf{SZKP}$. This last implication is considered unlikely as it would imply that the polynomial hierarchy collapses, contradicting Conjecture 1.2.1.

Second, we extend the techniques from Karp reductions to prove that if there exists a black-box reduction from **NP**-hardness to learning using constant rounds of adaptivity, then there exists an efficient-oracle reduction from **NP**-hardness to the existence of AIOWF using constant rounds of adaptivity. Such a reduction thus also implies that Pessiland and Minicrypt collapse.

Third, we study strongly black-box reduction from **NP**-hardness to learning, where the reduction accesses the hypothesis returned by the learning oracle only as a black-box. We show that if there exists a strongly black-box reduction from **NP**-hardness to learning, then there exists a black-box reduction from **NP**-hardness to inverting AIOWF (without restrictions on the adaptivity of the reduction). Then, we prove that if there exists a strongly black-box *non-adaptive* reduction from **NP**-hardness to learning, then $\mathbf{NP} \subseteq \mathbf{coAM}$ and therefore the polynomial hierarcy collapses to the second level [BHZ87], contradicting Conjecture 1.2.1.

Strategy

Consider a reduction R from **NP**-hardness to learning. R gets access to a learning oracle, and in order to use the learning oracle in a meaningful way, R must produce

a distribution of labelled examples which it feeds to the learning oracle. However, since R must be efficient, this means that the distribution of labelled examples must be efficiently samplable.

The key observation we will use is that because the distribution of labelled examples is efficiently samplable, one can convert R into a reduction R' that uses an oracle solving CircCons and CircLearn (defined in Definition 3.4.1, Definition 3.4.2) rather than an oracle that successfully tests PAC consistency or successfully PAC learns. Then we can use the facts we have learned from previous chapters to show that reducing an **NP**-complete language to CircCons or CircLearn implies that one can also reduce an **NP**-complete language to inverting AIOWF. Finally, we use standard results about zero knowledge and OWF (extended to the setting of AIOWF) to attain the surprising and/or implausible conclusions stated above.

5.1 Karp reductions

We first prove our results for the simple case of Karp reductions (Definition 1.3.1). In our setting, a Karp reduction R takes the instance of L and produces a single efficiently samplable distribution of labelled examples (X, Y). The reduction should guarantee that if $z \in L$, then (X, Y) is efficiently learnable, namely there exists a function $f \in \mathsf{SIZE}(n^2)$ such that $Y = f(X)$. On the other hand, if $z \notin L$ then the labelling (X, Y) should be far from all efficiently computable functions, namely $\mathsf{err}((X, Y), \mathsf{SIZE}(n^{\log n}))$ should be large, say larger than some $\beta \geq 1/\mathrm{poly}(n)$.

Notice this is exactly the notion of testing PAC β-consistency defined in Definition 3.1.2. Indeed, since Karp reductions are supposed to reduce one decision problem to another decision problem (not a search problem), by "Karp reduction to learning"

we actually mean a Karp reduction to testing PAC consistency. All of the known **NP**-hardness results for proper PAC learning are of this type [PV88, BR92, HJLT96, BDEL03, ABF+04, Fel06a, Fel06b, GR06, GKS07]. Although it is conceivable that one could Karp-reduce a language to a different decision problem associated with learning, testing consistency seems the most natural and is the only one that has been previously studied in the literature.

To make our results more general, we will consider an *agnostic* version of testing consistency.

Definition 5.1.1 (Testing agnostic (α, β) consistency)**.** *A tests for agnostic (α, β)-consistency of a concept class F if given access to any distribution of labelled examples (X, Y) the following holds.*

- YES instance: if $\mathsf{err}((X, Y), F) \leq \alpha$ then A outputs 1 with probability $1 - 2^{-n}$.
- NO instance: if $\mathsf{err}((X, Y), \mathsf{SIZE}(n^{\log\log n})) > \beta$, then A outputs 0 with probability $1 - 2^{-n}$.

Clearly, testing PAC β-consistency is the same as testing agnostic $(0, \beta)$ consistency.

Definition 5.1.2. *A reduction R is a Karp reduction from L to testing agnostic (α, β)-consistency if given any instance z and random coins ω, $R(z; \omega)$ generates a set of labelled examples sampled from a distribution (X, Y) such that:*

- *If $z \in L$, then (X, Y) is a YES instance of agnostic (α, β)-consistency*
- *If $z \notin L$, then (X, Y) is a NO instance of agnostic (α, β)-consistency.*

Furthermore, R runs in time poly(n).

The main observation is that because R is efficient, the distribution (X, Y) must be efficiently samplable. Therefore we can write without loss of generality that $R(x)$ outputs a circuit C sampling a distribution (X, Y) satisfying the above constraints. But this is exactly the notion of $\mathsf{CircCons}_{\alpha,\beta}$ that we defined earlier in Definition 3.4.1!

Lemma 5.1.3. *Suppose there exists a Karp reduction R from L to testing agnostic (α, β)-consistency. Then there exists a Karp reduction R' from L to $\mathsf{CircCons}_{\alpha,\beta}$.*

Proof. R' outputs the circuit C_z that computes $C_z(\omega) = R(z; \omega)$. By definition, $C_z(\omega)$ outputs a set of labeled examples that satisfy the promise of $\mathsf{CircCons}_{\alpha,\beta}$, and $|C_z| \leq \mathrm{poly}(n)$ because R is efficient. ■

Therefore, we can deduce the following theorem.

Theorem 5.1.4. *Suppose there is a Karp reduction R from L to testing agnostic (α, β)-consistency. Then:*

1. *If $\alpha < \beta - 1/\mathrm{poly}(n)$, then there is a non-adaptive efficient-oracle reduction from L to inverting AIOWF.*
2. *If $(1-2\alpha)^2 > (1-2\beta)$, then $L \in \mathbf{SZKA}$.*
3. *If $(1-2\alpha)^2 > (1-2\beta)$, and furthermore R maps $z \notin L$ to distributions (X, Y) such that $\mathsf{err}((X, Y), \mathcal{F}_{\mathsf{all}}) > \beta$, then $L \in \mathbf{SZKP}$.*

Proof. First apply Lemma 5.1.3 to get a reduction from L to $\mathsf{CircCons}_{\alpha,\beta}$. Let $C = R'(z)$, where C samples a distribution (X, Y).

Item 1 follows from the fact that an AIOWF inverter can be used to decide $\mathsf{CircCons}_{\alpha,\beta}$ (Corollary 3.5.2, Proposition 3.4.4).

Item 2 follows from the fact that $\mathsf{CircCons}_{\alpha,\beta} \in \mathbf{SZKA}$ for $(1-2\alpha)^2 > (1-2\beta)$ (Theorem 4.3.7).

Item 3 follows from the fact that this stronger NO condition actually satisfies the definition for $\mathsf{StatCircCons}_{\alpha,\beta}$, which we know is in **SZKP** (Lemma 4.3.3).

■

5.1.1 Extensions and corollaries

We can extend Item 1 of Theorem 5.1.4 to show that a Karp reduction from L to testing (α,β)-consistency would lead to surprising consequences.

Corollary 5.1.5. *Suppose there exists a Karp reduction from L to testing agnostic (α,β)-consistency. Then if L is hard on average, then there exists OWF.*

This corollary is proven by combining Item 1 with Lemma 5.2.6, which says that a reduction from (the worst-case hardness of) L to inverting AIOWF also gives a reduction that uses the average-case hardness of L to construct one-way functions. We prove this lemma later in Section 5.2.2 in the general, adaptive case.

Consider Corollary 5.1.5 in the case when $L = \mathsf{SAT}$. It is currently unknown how to build a reduction that uses average-case hardness of SAT to construct one-way functions, and in fact this is known as the "Pessiland vs. Minicrypt" problem in Impagliazzo's hierarchy of cryptographic hardness [Imp95]. Therefore Corollary 5.1.5 says that a Karp reduction from SAT to testing agnostic (α,β)-consistency would solve a long-standing open problem in cryptography. We will see in the next section that this is a more general phenomenon that also happens with black-box reductions with limited adaptivity.

Item 2 of Theorem 5.1.4 says that if furthermore $(1-2\alpha)^2 > (1-2\beta)$ then $\mathbf{NP} \subseteq$

SZKA, which we know holds assuming the existence of one-way functions [NOV06, HR07] but we do not know how to prove unconditionally.

Item 3 of Theorem 5.1.4 implies:

Corollary 5.1.6. *There is no Karp reduction from an* **NP***-complete language L to testing agnostic consistency where the NO condition is statistical unless* $\mathbf{NP} \subseteq \mathbf{SZKP}$ *(and thus Conjecture 1.2.1 is false).*

This is because Item 3 would then imply that $\mathbf{NP} \subseteq \mathbf{SZKP} \subseteq \mathbf{AM} \cap \mathbf{coAM}$, which by Theorem 1.2.2 implies that the polynomial hierarchy collapses, contradicting Conjecture 1.2.1. We will see in Section 5.3 that the hierarchy also collapses for strongly-black-box non-adaptive black-box reductions.

5.1.2 Truth table reductions

In fact, the results above can be strengthened to encompass not only Karp reductions but also monotone-$\mathbf{NC}^1$ reductions, which we define here. Consider variables taking value in $\{0, 1, \star\}$. We extend standard boolean algebra so that $\neg\star = \star$, $\star \wedge 1 = \star \wedge \star = \star$, $\star \wedge 0 = 0$, $\star \vee 1 = 1$, $\star \vee 0 = \star \vee \star = \star$. For a promise problem Π, let

$$\chi_\Pi(x) = \begin{cases} 1 & x \in \Pi_Y \\ 0 & x \in \Pi_N \\ \star & \text{else} \end{cases}$$

Definition 5.1.7. A promise problem Π reduces to a promise problem Γ via a (polynomial-time) *truth-table reduction* if there exists an efficient (possibly randomized) algorithm R taking an instance x of Π and some random bits and outputting a polynomial-size circuit C and instances $y_1, \ldots, y_k$ of Γ such that over the probability

of the reduction R:

$$x \in \Pi \Leftrightarrow \Pr[C(\chi_\Gamma(y_1), \ldots, \chi_\Gamma(y_k)) = 1] \geq 2/3$$
$$x \notin \Pi \Leftrightarrow \Pr[C(\chi_\Gamma(y_1), \ldots, \chi_\Gamma(y_k)) = 1] \leq 1/3$$

If the circuit C that R outputs is always an $\mathbf{NC}^1$ circuit (resp. monotone $\mathbf{NC}^1$ circuit), then the reduction is called $\mathbf{NC}^1$ (rep. monotone $\mathbf{NC}^1$) truth-table reduction.

Although the above definition is for promise problems, it extends in the obvious way to reductions from a promise problem Π to testing agnostic consistency (*i.e.* instead of generating instances y_i of a promise problem Γ, the reduction R generates instances (*i.e.* distributions of labeled examples) (X_i, Y_i) of agnostic consistency, and the rest of the definition is identical).

Theorem 5.1.4 can be extended to the case of monotone $\mathbf{NC}^1$ reductions:

Theorem 5.1.8. *Suppose there is a monotone* $\mathbf{NC}^1$ *truth-table reduction* R *from* L *to testing agnostic* (α, β)*-consistency. Then:*

1. *If* $\alpha < \beta - 1/\mathrm{poly}(n)$*, then there is a non-adaptive efficient-oracle reduction from* L *to inverting AIOWF.*
2. *If* $(1-2\alpha)^2 > (1-2\beta)$*, then* $L \in \mathbf{SZKA}$*.*
3. *If* $(1-2\alpha)^2 > (1-2\beta)$*, and furthermore* R *maps* $z \notin L$ *to distributions* (X, Y) *such that* $\mathsf{err}((X, Y), \mathcal{F}_{\mathsf{all}}) > \beta$*, then* $L \in \mathbf{SZKP}$*.*

Proof. First, for the same reason as Lemma 5.1.3, there exists a monotone $\mathbf{NC}^1$ truth-table reduction R' from L to $\mathsf{CircCons}_{\alpha,\beta}$.

Item 1 follows because again from the fact that an AIOWF inverter can be used to decide $\mathsf{CircCons}_{\alpha,\beta}$ (Corollary 3.5.2, Proposition 3.4.4).

Item 2 follows from the fact that $\mathsf{CircCons}_{\alpha,\beta} \in \mathbf{SZKA}$ for $(1-2\alpha)^2 > (1-2\beta)$ (Theorem 4.3.7), and also the fact that **SZKA** is closed under monotone $\mathbf{NC}^1$ truth-table reductions [Vad04, OV07].

Item 3 follows from the fact that this stronger NO condition actually satisfies the definition for $\mathsf{StatCircCons}_{\alpha,\beta}$, which we know is in **SZKP**, and also from the fact that **SZKP** is closed under monotone $\mathbf{NC}^1$ truth-table reductions [DSDCPY98, SV97].

■

The analogous extensions and corollaries Corollary 5.1.5 and Corollary 5.1.6 also hold for such truth-table reductions.

5.2 Black-box reductions

While Karp reductions cover the previously known **NP**-hardness results for *proper* learning, they still form a very restricted class of reductions. Phrased in terms of oracles, a Karp reduction takes an instance of L, produces from it a single distribution (X, Y), and asks the learning oracle to learn on this single instance. Even this query is only used to test consistency and does not use the actual hypothesis that the learning oracle returns in any meaningful way. Here we consider more powerful classes of reductions that can make many queries, as well as use the hypotheses that the oracle returns in a more meaningful way.

As defined in Section 1.3, a black-box reduction is allowed to generate many queries for its learning oracle and may examine the hypotheses that the learning oracle returns. It may even adaptively generate new queries depending on answers the oracle gave

to previous queries.

Definition 5.2.1. A reduction R that decides L using an agnostic learning oracle if R makes queries of the following form. For each query, R constructs a distribution (X, Y), samples a set $S = \{(x_i, y_i)\}_{i \leq \text{poly}(n)}$ of independent labeled examples according to (X, Y), and passes S to the oracle. The distribution (X, Y) can depend on the instance z to be decided, the random coins ω, and the hypotheses that R received from the oracle in response to queries from previous adaptive rounds. Given access to any oracle $\mathcal{O}$ that agnostically learns $\mathsf{SIZE}(n^2)$ circuits, with probability $1 - 2^{-n}$ the reduction $R^{\mathcal{O}}(z)$ outputs 1 if $z \in L$ and outputs 0 if $z \notin L$.

Since R is efficient, each query (X, Y) must be efficiently samplable. This suggests the following lemma.

Lemma 5.2.2. *For any $c = \text{poly}(n)$, if there exists a c-adaptive black-box reduction R from L to agnostic learning, then there exists a c-adaptive black-box reduction R' from L to* CircLearn.

Proof. R' samples random coins ω and emulates the execution of $R(z; \omega)$ as follows. R' constructs and evaluates the Cook-Levin tableau of the execution of $R(z; \omega)$ until it hits a round of oracle queries. When R makes a query (X, Y) to the agnostic learning oracle, R' takes the subcircuit of the tableau that samples labeled examples according to (X, Y), call this circuit C, and queries its CircLearn oracle with C. R' then writes the hypothesis that it receives back into the tableau and continues the execution of R, still keeping track of the tableau so that it may be used for the next oracle query.

It follows that for every query (X, Y) that R makes, since R' makes a query for a circuit C sampling (X, Y), R' gets back from its CircLearn oracle with high probability

a hypothesis h such that

$$\mathsf{err}((X,Y),h) \leq \mathsf{err}((X,Y),\mathsf{SIZE}(n^2)) + \varepsilon$$

Namely, if R' is given access to an oracle $\mathcal{O}$ that solves $\mathsf{CircLearn}$, then when R' emulates the execution of R, from R's point of view it gets access to an oracle $\mathcal{O}'$ that agnostically learns $\mathsf{SIZE}(n^2)$. Therefore:

$$\Pr[(R')^{\mathcal{O}}(z;\omega) = L(z)] = \Pr[R^{\mathcal{O}'}(z;\omega) = L(z)] \geq 1 - 2^{-n}$$

where the last inequality follows from the correctness of R. Furthermore, from the construction of R' above it follows that the adaptivity of R' is identical to the adaptivity of R. ■

5.2.1 Constant-adaptive reductions

Combining Lemma 5.2.2 with Corollary 3.5.2, we obtain the following theorem. Notice that we can only handle $c = O(1)$ adaptive rounds.

Theorem 5.2.3. *For any $c = O(1)$, if there is a c-adaptive black-box reduction R from L to agnostic learning, then there exists a c-adaptive efficient-oracle reduction R' from L to inverting AIOWF.*

Proof. Let $R = (R_1, \ldots, R_c, M)$ be the decomposition of R into its c adaptive rounds, as described in Definition 1.3.5. Let A be the non-adaptive black-box reduction from $\mathsf{CircLearn}$ to inverting AIOWF that we obtain from Corollary 3.5.2.

Let I be an AIOWF inverting oracle. Naively we could try to simply compose R with A, but this does not work. Notice that R needs to write down the hypothesis that it obtains from its learning oracle, but the hypothesis that A returns contains

embedded calls to I. Therefore, if I is inefficient, there is no way that R could write down the hypothesis that it got back from A.

This necessitates the assumption that I be efficient. If I is efficient, then even though the hypothesis h contains embedded calls to I, it can still be represented as a polynomial-size circuit, and so R could write this hypothesis into the tableau in polynomial time and continue execution.

Suppose that R_j makes at most $m(n,q) = \text{poly}(n,q)$ queries in the j'th round of queries, assuming that the largest hypothesis received from the CircLearn oracle in rounds $1,\ldots,j-1$ has size at most $q(n)$. For each $j \in [c]$, let $C^{\omega}_{j,1},\ldots,C^{\omega}_{j,m}$ be the queries that R_j generates. Let D_j be the circuit computing $D_j(\omega,i,r) = (\omega,i,C^{\omega}_{j,i}(r))$. The circuit D_j can depend on the input z, and the hypotheses $h_{j',i'}$ for $j' < j$ and $i' \in [m]$ that R_j got in response to queries from previous rounds. Let $p(n,q) = \text{poly}(n,q)$ bound the size of D_j, where q is an upper bound on the size of hypotheses $h_{j',i'}$ returned by the oracle in previous rounds.

By Remark 2.3.9, the existence of an efficient AIOWF inverter implies the existence of an efficient I satisfying for all circuits D of size $|D| \leq s$ such that $D(\omega,i,r) = (\omega,i,x)$

$$\Delta((\omega,i,r,x),(I(D,\omega,i,x),x)) \leq O(\varepsilon^6/(scmn))$$

where ω,i,r are uniformly random.

In the reduction, we will use the inverting oracle I along with the procedure A from Lemma 3.5.1 that given a circuit sampling a distribution of labeled examples, produces from I a hypothesis for that circuit.

The reduction: our reduction R' does the following:

1. Sample random coins ω, and use the input z and the random coins ω to construct the query circuit D_1 that corresponds to the first round.

2. Use the distributional inverter I along with the learning procedure A on the sampling circuit $D_1(\omega, i, \cdot)$ (*i.e.* for fixed ω and i but random r) for each $i \in [m]$ in order to get hypotheses $h^{\omega}_{1,i}$ in the first round.

3. Inductively, use ω, z and the hypotheses received in previous rounds $1, \ldots, j-1$ to construct D_j and use the inverting oracle I and the procedure A on the circuit $D_j(\omega, i, \cdot)$ to get hypotheses $h^{\omega}_{j,i}$ for all $i \in [m]$ in the j'th round.

4. At the last round, apply M to ω, z and all the previously the obtained hypotheses to decide whether $z \in L$.

Correctness: We first prove the following lemma, which says that if one can distributionally invert a circuit $D(\omega, i, r)$ well enough, where $\omega \in \{0,1\}^{\mathrm{poly}(n)}, i \in [\mathrm{poly}(n)], r \in \{0,1\}^{\mathrm{poly}(n)}$, then with high probability over ω, one can distributionally invert $D(\omega, i, \cdot)$ simultaneously for all i.

Lemma 5.2.4. *For any $m, c, s = \mathrm{poly}(n)$, suppose for all circuits D of size at most s taking inputs $\omega \in \{0,1\}^{\mathrm{poly}(n)}, i \in [m], r \in \{0,1\}^{\mathrm{poly}(n)}$ and outputting $D(\omega, i, r) = (\omega, i, x)$ where $x \in \{0,1\}^n$ that the inverter I satisfies the following:*

$$\Delta((\omega, i, r, x), (I(D, \omega, i, x), x)) \leq O(\varepsilon^6/(scmn))$$

where the randomness is over uniform ω, i, r and the random coins of I.

Then for all D, with probability $1 - \frac{1}{cn}$ we get a good ω, namely ω satisfies that for all $i \in [m]$,

$$\Delta((\omega, i, r, x), (I(D, \omega, i, x), x)) \leq O(\varepsilon^6/s)$$

where the randomness is only over the choice of r and the random coins of I.

Proof. The lemma is proven by an averaging argument. With probability at least

$1 - \frac{1}{cn}$ we get a "good" ω, where it holds that

$$\Delta((\omega, i, r, x), (I(D, i, x), x)) \leq O(\varepsilon^6/(sm))$$

where now ω is *fixed* and the randomness is over r, i and the random coins of I. By averaging we have that for good ω and for all $i \in [m]$,

$$\Delta((\omega, i, r, x), (I(D, \omega, i, x), x)) \leq O(\varepsilon^6/s)$$

where now both ω, i are fixed, and the randomness is only over r and the coin tosses of I. ■

Consider the first round queries of the reduction generated by D_1. It holds that $|D_1| \leq s_1(n)$ for some polynomial $s_1(n)$ (since the first round of queries does not depend on any oracle responses). Say that $s_1(n) \leq n^a$.

Applying Lemma 5.2.4 to D_1, we get with probability $1 - \frac{1}{cn}$ a good ω. In this case, we have for all $i \in [m]$ that

$$\Delta((\omega, i, r, x), (I(D_1, \omega, i, x), x) \leq O(\varepsilon^6/s_1)$$

Therefore, since $|C^{\omega}_{1,i}| \leq |D_1|$, we are in the setting of Lemma 3.5.1, which says that given a sufficiently good inverter we can learn any circuit. We may therefore apply the reduction A in that lemma, giving it oracle access to I, in order to get with probability $1 - 2^{-n}$ hypotheses $h^{\omega}_{1,i}$ that satisfy

$$\mathsf{err}((X^{\omega}_{1,i}, Y^{\omega}_{1,i}), h^{\omega}_{1,i}) \leq \mathsf{err}((X^{\omega}_{1,i}, Y^{\omega}_{1,i}), \mathcal{F}_{\mathsf{all}}) + \varepsilon \leq \mathsf{err}((X^{\omega}_{1,i}, Y^{\omega}_{1,i}), \mathsf{SIZE}(n^2)) + \varepsilon$$

where $(X^{\omega}_{1,i}, Y^{\omega}_{1,i})$ is the distribution sampled by $C^{\omega}_{1,i}$. Let $q(s_1)$ be an upper bound on the size of the hypotheses thus obtained using A, *i.e.* for all i, $|h^{\omega}_{1,i}| \leq q(s_1)$.

Going from D_{j-1} to D_j: repeat this procedure inductively: for each $j \in [c]$, we have that the previous rounds gave us hypotheses of size at most $q(s_{j-1})$, therefore

$|D_j| \leq p(n, q(s_{j-1}))$. Set $s_j(n) = p(n, q(s_{j-1}(n)))$. Again we have by Lemma 5.2.4 an inverter I that inverts the i'th query in round j, so we can again run the learning algorithm A of Lemma 3.5.1 with this inverter in order to get back ε-hypotheses $h^{\omega}_{j,i}$ of size at most $q(s_j(n))$.

Correctness of this reduction follows from the fact that ω is good for each round with probability $1 - \frac{1}{cn}$, and therefore it is good for all rounds simultaneously with probability $1 - \frac{1}{n}$. Given a good ω, the learning procedure A with oracle access to I correctly solves CircLearn for all rounds and all queries, and so by the correctness of R, the reduction R' outputs the correct answer with $1 - 2^{-n}$. Therefore, the overall error probability of R' is $1 - \frac{1}{n} - 2^{-n}$, which can be amplified by repetition.

Efficiency: it remains to check that R' is efficient if $c = O(1)$. Suppose $p(n, q) \leq (nq)^a$ (we assumed also that $s_1(n) \leq n^a$, but this is fine since we could just take the minimal a that satisfies both). Suppose that $q(s) \leq n^b$ (since $q(s)$ is the size of the hypothesis returned and contains the code of I, b is constant if and only if the oracle I is efficient). We claim that:

$$s_i \leq n^{a^i b^i + a(a^{i-1}b^{i-1}-1)/(ab-1)}$$

This holds trivially for $i = 1$, and we check inductively for $i \geq 2$ that

$$\begin{aligned}
s_i(n) &= p(n, q(s_{i-1})) \\
&\leq (n s_{i-1}^b)^a \\
&\leq n^{a + ab(a^{i-1}b^{i-1} + a(a^{i-2}b^{i-2}-1)/(ab-1))} \\
&\leq n^{a^i b^i + a(a^{i-1}b^{i-1} - ab)/(ab-1) + a} \\
&\leq n^{a^i b^i + a(a^{i-1}b^{i-1}-1)/(ab-1)}
\end{aligned}$$

Therefore, as long as R runs for a total of $c = O(1)$ adaptive rounds, the largest circuits that R' ever sees have size at most $s_c(n)$, and the largest hypotheses it sees

have size at most $q(s_c(n))$, so overall R' still runs in time $\text{poly}(q(s_c(n))) = \text{poly}(n)$ time.

■

Definition of reduction to learning

As with Definition 5.1.2 of Karp reductions from a decision problem to learning, one could argue that our notion of black-box reduction (Definition 5.2.1) is not sufficiently general. Our definition requires a query to a learning oracle to be a set of labeled examples drawn independently from a distribution (X, Y). One could consider a more general notion, where the reduction does not explicitly construct a distribution (X, Y) but instead generates directly a set of labeled examples. Clearly our notion also satisfies this second notion, although the converse may not hold.

However, we argue (informally) that reductions under the second notion cannot be more powerful than reduction under our notion. If the set of labeled examples are not drawn independently from some distribution (X, Y), then the input does not satisfy the promise that the learning oracle requires, so there is no reason that the learning oracle should produce a useful hypothesis in response. In particular, different learning oracles could respond with different arbitrary hypotheses to such inputs violating the promise. In such a case, if the reduction were to always succeed given a learning oracle, it must in some sense be solving the problem without using the power of the learning oracle.

5.2.2 Collapsing Pessiland and Minicrypt

Theorem 5.2.3 gives as a corollary a reduction from *average-case hardness* of L to the existence of *standard* one-way functions. In Impagliazzo's taxonomy of worlds

[Imp95], a world where average-case hardness of **NP** exists is called "Pessiland" and a world in which one-way functions exist is called "Minicrypt". It is not known whether the two are in fact the same, or whether **NP** can be hard on average and yet one-way functions do not exist.

Corollary 5.2.5. *Suppose there is a constant-adaptive black-box reduction R from L to agnostic learning. Then if L is hard on average, then there exists OWF.*

This corollary follows from Theorem 5.2.3 and the following lemma, which says that a reduction from (the worst-case hardness of) L to inverting AIOWF also gives a reduction that uses the average-case hardness of L to build OWF.

Lemma 5.2.6. *Suppose there exists a c-adaptive efficient-oracle reduction R that decides L given access to an AIOWF inverter for some $c = \text{poly}(n)$. Then if there exists a hard-on-average distribution Z for L, then there exists OWF.*

Proof. Recall that if Z is hard on average for L, this means Z is samplable by a uniform family of polynomial-size circuits (which for simplicity we also call Z), and there exists $\delta_0(n) \geq 1/\text{poly}(n)$ such that for all efficient algorithms A,

$$\Pr_{z \leftarrow_R Z, A}[A(z) = L(z)] \leq 1 - \delta_0(n)$$

Let $z = Z(\zeta)$ denote the sample drawn from Z using random coins ζ.

Let $R = (R_1, \ldots, R_c, M)$ be the decomposition of R into its adaptive rounds. Let $1-\delta_R(n)$ denote the success probability that R requires of its inverting oracle in order to guarantee that it decides L correctly with probability $1 - 2^{-n}$.

Let $D_j(z, \omega, i, r, \vec{a})$ be the circuit that outputs $(z, \omega, i, C_{j,i}^{z,\omega}(r))$ where z is an instance of L, ω are random coins for R, $i \in [m]$, and r are random coins for sampling $C_{j,i}^{z,\omega}$, where $C_{j,1}^{z,\omega}, \ldots C_{j,m}^{z,\omega}$ are the circuits that R_j queries its inverting oracle in the j'th

round given z, ω, and $\vec{a}$, which are answers to all queries from previous rounds. Notice that each D_j is uniformly computable, *i.e.* it is completely defined by $R_1, \ldots, R_j$.

We will now prove by induction on (a slight modification of) the D_j that DOWF exist (and therefore by Theorem 2.3.5 OWF also exist).

Define $D'_1(\zeta, \omega, i, r) = D_1(Z(\zeta), \omega, i, r)$. If D'_1 is a DOWF then we are done. So suppose not, set $\delta_{\mathsf{min}} = \min(\delta_R(n), \delta_0(n))$, and set $\delta'(n) = (\frac{\delta_{\mathsf{min}}}{2mc})^2/m$. Since D'_1 is not a DOWF, then there exists an efficient I_1 satisfying

$$\Delta((\zeta, \omega, i, r), I_1(D'_1(\zeta, \omega, i, r); \rho_1)) \leq \delta'(n) \tag{5.2.1}$$

where the randomness is over ζ, ω, i, r and ρ_1, which is the coin tosses of I_1.

Inductively, let D'_j be the circuit that computes $D'_j(\zeta, \omega, i, r, \vec{\rho}) = D_j(Z(\zeta), \omega, i, r, \vec{a})$, *i.e.* D'_j computes a random query from the j'th round of queries. Here $\vec{\rho} = (\rho_1, \ldots, \rho_{j-1})$ are random coins such that for each $j' \in [j-1]$, $\rho_{j'}$ contains enough random coins to invoke $I_{j'}$ m independent times, and $\vec{a}$ are the inverses for rounds $1, \ldots, j-1$ that are obtained using $\vec{\rho}$ as random coins. Since $I_1, \ldots, I_{j-1}$ are efficient by the inductive assumption, therefore D'_j is efficient. If D'_j is a DOWF then we are done, otherwise there exists an inverter I_j such that

$$\Delta((\zeta, \omega, i, r), I_j(D'_j(\zeta, \omega, i, r; \vec{\rho}); \rho_j)) \leq \delta'(n)$$

But at least one of the D'_j for $j \in [c]$ must be a DOWF, since otherwise D'_c could be used to decide L correctly with high probability. Namely, if D'_c is not a DOWF, then there exists I_c satisfying Inequality 5.2.1 for the c'th round. We will show that this implies an efficient method, using the efficient inverters $I_1, \ldots, I_c$, that decides whether or not $z \in L$.

To prove this, define $(\zeta, \omega, \vec{\rho})$ to be δ-good for I_j if for all $i \in [m]$,

$$\Delta((\zeta, \omega, i, r), I_j(D'_j(\zeta, \omega, i, r, \vec{\rho}); \rho_j)) \leq \delta$$

where the randomness is *only over* r and the random coins ρ_j of I.

We will apply the following averaging argument:

Claim 5.2.7. *For any D and any $\delta(n) \geq 1/\mathrm{poly}(n)$, suppose there exists an efficient I such that*

$$\Delta((\zeta, \omega, i, r), I(D(\zeta, \omega, i, r, \vec{\rho}); \rho)) \leq \frac{\delta^2}{m}$$

over random $\zeta, \omega, i, r, \vec{\rho}, \rho$. Then, with probability $1 - \delta$, $(\zeta, \omega, \vec{\rho})$ is δ-good for I.

Proof. The claim is proven by an averaging argument. Given the hypothesis:

$$\Delta((\zeta, \omega, i, r), I(D(\zeta, \omega, i, r, \vec{\rho}); \rho)) < \frac{\delta^2}{m}$$

where the randomness is over $\zeta, \omega, i, r, \vec{\rho}, \rho$. Applying Markov's inequality shows that with probability $1 - \delta$ over $(\zeta, \omega, \vec{\rho})$

$$\Delta((\zeta, \omega, i, r), I(D(\zeta, \omega, i, r, \vec{\rho}); \rho)) < \frac{\delta}{m}$$

for fixed $\zeta, \omega, \vec{\rho}$ and random r, ρ. By averaging, for each i it holds that

$$\Delta((\zeta, \omega, i, r), I(D(\zeta, \omega, i, r, \vec{\rho}); \rho)) < \delta$$

which implies that $\zeta, \omega, \vec{\rho}$ is δ-good for I. ■

Applying this claim to our setting, suppose that none of the D'_j are DOWF, namely for each $j \in [m]$, there exists an inverter I_j such that

$$\Delta((\zeta, \omega, i, r), I_j(D'_j(\zeta, \omega, i, r; \vec{\rho}); \rho_j)) \leq \delta'(n)$$

where as we defined previously $\delta'(n) = (\frac{\delta_{\mathsf{min}}}{2mc})^2/m$ and $\delta_{\mathsf{min}} = \min(\delta_0(n), \delta_R(n))$. We deduce from Claim 5.2.7 that for each $j \in [m]$ with probability $1 - \frac{\delta_{\mathsf{min}}}{2mc}$ it holds that $(\zeta, \omega, \vec{\rho})$ is $\frac{\delta_{\mathsf{min}}}{2mc}$-good for I_j, and so therefore by a union bound, with probability $1 - \frac{\delta_{\mathsf{min}}}{2m}$ it holds that $(\zeta, \omega, \vec{\rho})$ is $\frac{\delta_{\mathsf{min}}}{2mc}$-good for all I_j simultaneously.

Define the distribution $Q_j = (z, \omega, \vec{a}_j)$ where $z \leftarrow_{\mathrm{R}} Z$, ω is uniform and $\vec{a}_j$ is a vector of inverses to all of the queries that D_j asks in rounds $1, \ldots, j$. In Q_j, these inverses $\vec{a}$ are generated using the inverters $I_1, \ldots, I_j$ with random coins $\vec{\rho}$. Define $Q_j^{\mathsf{ideal}} = (z, \omega, \vec{a}_j)$ identically, except the answers $\vec{a}_j$ are generated by an ideal inverter I^{ideal} that samples from the exact conditional distribution of preimages.

Claim 5.2.8. *For any $\delta > 0$, conditioned on $(\zeta, \omega, \vec{\rho})$ being δ-good for $I_1, \ldots, I_j$, it holds that $\Delta(Q_j, Q_j^{\mathsf{ideal}}) \leq jm\delta$.*

Proof.

- For $j = 1$, it holds from the triangle inequality because there are at most m queries in the first round and each one deviates by at most δ.
- Assume it holds for $j-1$, then since the queries for the j'th round are computed from Q_{j-1}, this means by the inductive hypothesis that the distribution of queries is at most $(j-1)m\delta$-far in Q_j versus Q_j^{ideal}. Since $(\zeta, \omega, \vec{\rho})$ is good for I_j and there are at most m queries in the j'th round, it therefore follows that this introduces at most an additional $m\delta$ to the statistical difference.

■

This implies that conditioned on $(\zeta, \omega, \vec{\rho})$ being $\frac{\delta_{\mathsf{min}}}{2mc}$-good, it holds that $\Delta(Q_c, Q_c^{\mathsf{ideal}}) \leq \frac{\delta_{\mathsf{min}}}{2}$. But Q_c completely determines an execution of R on a random input $z \leftarrow_{\mathrm{R}} Z$, since Q_c contains (1) ζ which describes the input $z = Z(\zeta)$, (2) ω which determines the random tape of the reduction, (3) all the oracle responses for all rounds.

Let M be the efficient verification procedure from the reduction R that takes such a complete execution and decides whether to accept or reject. Let G be the event that

$(\zeta, \omega, \vec{\rho})$ are good, then

$$\begin{aligned}
&\Delta((Q_c, M(Q_c)),(Q_c^{\mathsf{ideal}}, M(Q_c^{\mathsf{ideal}}))) \\
&\qquad = \Pr[\overline{G}] \cdot \Delta((Q_c, M(Q_c) \mid \overline{G}), (Q_c^{\mathsf{ideal}}, M(Q_c^{\mathsf{ideal}}) \mid \overline{G})) \\
&\qquad\quad + \Pr[G] \cdot \Delta((Q_c, M(Q_c) \mid G), (Q_c^{\mathsf{ideal}}, M(Q_c^{\mathsf{ideal}}) \mid G)) \\
&\qquad \leq \tfrac{\delta_{\mathsf{min}}}{2m} + \Delta((Q_c, M(Q_c) \mid G), (Q_c^{\mathsf{ideal}}, M(Q_c^{\mathsf{ideal}}) \mid G)) \\
&\qquad < \delta_{\mathsf{min}}(\tfrac{1}{2} + \tfrac{1}{2m}) \\
&\qquad < \tfrac{3}{4}\delta_{\mathsf{min}}
\end{aligned}$$

Running M on Q_c is identical to running the reduction R on $z \leftarrow_{\mathrm{R}} Z$ using the inverting oracle I_j in the j'th round. This is efficient, since we assumed all the I_j are efficient. From the above fact that $(Q_c, M(Q_c))$ and $(Q_c^{\mathsf{ideal}}, M(Q_c^{\mathsf{ideal}}))$ are $\frac{3}{4}\delta_{\mathsf{min}}$-close

$$\Pr[R^{I_1,\ldots,I_c}(z) = L(z)] = \Pr_{Q_c}[M(Q_c) = L(z)] > \Pr_{Q_c^{\mathsf{ideal}}}[M(Q_c^{\mathsf{ideal}}) = L(z)] - \tfrac{3}{4}\delta_{\mathsf{min}}$$

Since Q_c^{ideal} is generated using a perfect distributional inverting oracle, by the correctness of R it follows that M decides correctly given Q_c^{ideal} with probability $1 - 2^{-n}$, which allows us to conclude that

$$\Pr[R^{I_1,\ldots,I_c}(z) = L(z)] \geq 1 - \tfrac{3}{4}\delta_{\mathsf{min}} - 2^{-n} \geq 1 - \delta_0(n)$$

But this contradicts the average-case hardness of L, so therefore at least one of the D'_j must be a DOWF. ■

Interpretation: if L is **NP**-complete, then such a reduction from the average-case hardness of L to inverting OWF would collapse two worlds in Impagliazzo's hierarchy, "Minicrypt" and "Pessiland". Our result highlights this connection between the hardness of learning with a well-studied question in complexity and cryptography.

Our result does not say that proving such a reduction is impossible or implausible, but rather it would, in the process, solve a longstanding open problem in cryptography.

Without giving much idea as to whether or not such a reduction exists, it nevertheless indicates the difficulty of proving such a reduction.

5.3 Strongly black-box reductions

Collapsing "Minicrypt" and "Pessiland" would be a great breakthrough, but it is not implausible. Indeed, common wisdom in complexity and cryptography says that not only does cryptography exist, but stronger forms of cryptography such as public-key encryption and oblivious transfer exist (which correspond to the "Cryptomania" world in Impagliazzo's hierarchy).

In this section we further restrict the reduction in the way it accesses the hypothesis returned by the learning oracle by requiring that the reduction access the returned hypotheses as black boxes and should decide L correctly regardless of how the hypotheses are implemented.

Definition 5.3.1. A reduction R is a strongly black-box reduction from L has the following form. R takes access to an $\mathcal{O}$ taking a set of labeled examples S and an additional unlabeled x. Each query R makes is of the following form: R constructs a distribution (X, Y), and generates $S = \{(x_i, y_i)\}_{i \leq \text{poly}(n)}$ consisting of polynomially many independent labeled examples drawn from (X, Y). Given $\mathcal{O}$ such that with probability $1 - 2^{-n}$ over the choice of examples S, $\mathcal{O}$ satisfies

$$\mathsf{err}((X, Y), \mathcal{O}(S, \cdot)) \leq \mathsf{err}((X, Y), \mathsf{SIZE}(n^2)) + \varepsilon$$

then with probability $1 - 2^{-n}$ $R^{\mathcal{O}}(z)$ outputs 1 if $z \in L$ and outputs 0 if $z \notin L$.

One can think of this is as an "information-theoretic" version of learning, since the learning oracle does not return a hypothesis to the reduction as a circuit but instead

gives the reduction oracle access to a good hypothesis. Thus, it is possible for the learning oracle to give access to inefficient hypotheses.

We will show that the existence of a strongly-black-box reduction of arbitrary polynomial adaptivity from **NP**-hardness to agnostic learning implies a black-box reduction from **NP**-hardness to inverting AIOWF, bypassing the constant-adaptivity restriction of Theorem 5.2.3. Furthermore, we show that the existence of a strongly-black-box non-adaptive reduction from **NP**-hardness to agnostic learning implies that $\mathbf{NP} \subseteq \mathbf{coAM}$, which contradicts Conjecture 1.2.1 and thus is considered not only surprising, but in fact implausible.

5.3.1 Strongly-black-box adaptive reductions

In the case of adaptive reductions, we can lift the limitation that $c = O(1)$ in Theorem 5.2.3.

Theorem 5.3.2. *For any $c = \mathrm{poly}(n)$, suppose there exists a strongly-black-box c-adaptive reduction R from deciding L to agnostic learning. Then there exists a black-box c-adaptive reduction R' from deciding L to inverting AIOWF.*

Proof. The proof is essentially identical to the proof of Theorem 5.2.3. We construct $D_1, \ldots, D_c$ as there, except that D_j does not depend on the code of hypotheses returned in previous rounds, it *only* depends on the input-output behavior of the hypotheses from previous rounds. Because of this, the complexity of R' does not depend on the complexity of the inverter I, and so the complexity of R' does not blow up in each round. This means that R' can successfully emulate the execution of R for $c = \mathrm{poly}(n)$ rounds in polynomial time. Furthermore, because R' only uses the input-output behavior of the hypotheses in turn only depends on the input-output

behavior of I, it holds that R' is black-box. ■

Since a black-box reduction is also efficient-oracle (but not necessarily vice versa), we can apply Lemma 5.2.6, which says that a reduction from deciding L in the worst case to inverting AIOWF also gives a reduction that uses the average-case hardness of L to build OWF. We can conclude that

Corollary 5.3.3. *For any $c = \text{poly}(n)$, suppose there exists a strongly-black-box c-adaptive reduction R from deciding L to agnostic learning. Then if L is hard on average, then there exists OWF.*

5.3.2 Strongly-black-box non-adaptive reductions

The following result says that non-adaptive strongly black-box reductions from **NP**-hardness to learning do not exist unless **NP** $\subseteq$ **coAM**.

Theorem 5.3.4. *Suppose there exists a non-adaptive strongly-black-box reduction from L to learning. Then $L \in$ **coAM**.*

This theorem is proved in two steps. First, we show that a non-adaptive strongly-black-box reduction from L to learning implies a special kind of non-adaptive reduction from L to inverting AIOWF. Then we adapt a result of [BT06, AGGM06] to show that such a special non-adaptive reduction from L to inverting AIOWF implies that $L \in$ **coAM**.

The special kind of reduction we use guarantees that for each input z, the reduction only queries the AIOWF to invert a single circuit and not multiple circuits.

Definition 5.3.5. A reduction R from L to inverting AIOWF is called fixed-auxiliary-input if for each z, there exists some circuit C such that all queries $R(z)$ makes to

the inverter I are for the same circuit C, namely $R(z)$ never makes two queries of the form $I(C, y), I(C', y')$ for $C \neq C'$.

Of course, which circuit C the reduction queries can vary depending on the input z.

Proof of Theorem 5.3.4. The proof of Theorem 5.3.4 follows from two lemmas:

Lemma 5.3.6. *Suppose there exists a non-adaptive strongly-black-box reduction R from L to learning. Then there exists a fixed-auxiliary-input non-adaptive black-box reduction R' from L to inverting AIOWF.*

Proof of Lemma 5.3.6. One simply needs to inspect the proof of Theorem 5.2.3 for the case that R is non-adaptive. Since the reduction R is non-adaptive, the querying circuit D_1 is the only circuit that R queries to the oracle, and so the reduction is fixed-auxiliary-input. Furthermore, as observed in Theorem 5.3.2, because R is strongly-black-box, the reduction R' obtained from Theorem 5.2.3 is black-box because R only uses the input-output behavior of the returned hypotheses. ■

The second step is to prove that a fixed-auxiliary-input non-adaptive black-box reduction from L to inverting AIOWF implies that $L \in \mathbf{coAM}$.

Lemma 5.3.7. *Suppose there exists a fixed-auxiliary-input non-adaptive black-box reduction from L to inverting AIOWF. Then $L \in \mathbf{coAM}$.*

Proof of Lemma 5.3.7. We apply the main idea of [FF93, BT06, AGGM06], which is to construct an **AM** protocol for the complementary language $\overline{L}$. Let $\overline{R}$ denote the reduction that is identical to R except it flips the answer, *i.e.* if R outputs the bit b, $\overline{R}$ outputs $1 - b$. Let C be the fixed auxiliary input for the reduction. The verifier in the protocol for $\overline{L}$ will execute the reduction $\overline{R}$, which is efficient except for the

inverter queries. To obtain answers to these inverter queries, the verifier sends the queries to the all-powerful prover.

By the correctness of $\overline{R}$, whenever $x \in \overline{L}$ the prover can respond to all queries with honest inverses, causing the verifier to accept. However, when $x \notin \overline{L}$ the prover could conceivably cheat by answering queries maliciously; for example, it could lie and say that a query y is not invertible even when it is, or it could adaptively choose the inverses in a way that adversely affects R's execution. To prevent such cheating, [AGGM06] uses techniques from [FF93, BT06] to force the prover to behave honestly.

We can in fact apply [AGGM06]'s proof verbatim to our setting except replacing their use of a OWF with our AIOWF and replacing calls to their OWF-inverting oracle to our AIOWF-inverting oracle (with fixed auxiliary input).

Because the proof of [AGGM06] is complex and outside the scope of this paper, we illustrate the reason [AGGM06] can be adapted to the setting of fixed-auxiliary-input reductions by discussing a special case. Let us assume that R is a non-adaptive fixed-auxiliary-input reduction from **NP**-hardness to inverting AIOWF where R's auxiliary input is a circuit for an *injective* function.

Namely, we assume that R makes one round of queries $(C, x_1), \ldots, (C, x_k)$ where C is the fixed auxiliary input mapping $\{0,1\}^p \to \{0,1\}^r$ in a one-to-one way (in particular, $r \geq p$). We will describe the protocol in this case and informally argue its correctness, referring the reader to [AGGM06] for full details as well as the proof of the general case. We will use the Goldwasser-Sipser lower-bound protocol [GS89] as well as the Fortnow upper-bound protocol [For87] (see also [AH91]), for which we include a reference in Appendix A.1.

Assume that all the x_i in R's single round of queries are identically (but not necessarily independently) distributed. This is without loss of generality because we can

randomly permute the queries of R to achieve this kind of query distribution. Let D denote the circuit mapping $\{0,1\}^q \to \{0,1\}^r$ that takes q random coins and samples from R's query distribution.

Heavy queries protocol: for an appropriate polynomial $\alpha(n)$, the verifier engages in a protocol with the prover to compute the probability that the reduction asks a heavy query. A heavy query y is one such that $\Pr[D(U_q) = y] \geq \alpha(n)2^{-p}$, *i.e.* it occurs with much larger probability under $D(U_q)$ than it does under $C(U_p)$. The heavy queries protocol computes $p_{\mathsf{heavy}} = \Pr[D(U_q) \text{ heavy}]$ by doing the following.

1. The verifier samples polynomially many $r_i \leftarrow_{\mathrm{R}} U_q$ and computes $y_i = D(r_i)$ and sends the y_i to the prover. The prover responds with claimed sizes s_i.

2. The verifier asks the prover to use the Goldwasser-Sipser lower-bound protocol to prove that $|D^{-1}(y_i)| \geq s_i$.

3. Using the hidden sample r_i, the verifier asks the prover to use the Fortnow upper-bound protocol to prove that $|D^{-1}(y_i)| \leq s_i$.

The verifier observes the fraction of the y_i that satisfy $s_i 2^{-q} \geq \alpha(n)2^{-p}$ and uses it as his estimate for p_{heavy}.

This protocol either gives a good approximation for p_{heavy} or aborts: if the prover tries to cheat in too many instances of the lower/upper bound protocols, the soundness guarantees of the lower/upper bound protocols ensure that with high probability, at least one of the upper or lower bound protocols will abort and cause the verifier to abort.

Hiding protocol: let D' denote the following distribution: sample a random element from $y \leftarrow_{\mathrm{R}} D$ conditioned on y not being heavy. The hiding protocol computes $p_{\mathsf{image}} = \Pr_{y \leftarrow_{\mathrm{R}} D'}[\exists x,\ C(x) = y]$. The protocol is as follows:

1. The verifier samples poly(n) many $r_i \leftarrow_{\mathrm{R}} U_q$ and computes $y_i = D(r_i)$. The verifier also samples poly(n) many $x_i \leftarrow_{\mathrm{R}} U_p$ and computes $y'_i = C(x_i)$. The number of y'_i sampled according to C should be much larger than the number of y_i sampled according to D.

2. The verifier randomly permutes the set of y_i, y'_i and call the set of queries $z_1, \ldots, z_k$. Let $S \subseteq [k]$ be the subset of $\{z_i\}$ that are drawn according to D.

3. The prover replies for each z_i with one of the following. First, if z_i is a heavy query, then the prover claims it is heavy and engages the verifier in a proof of heaviness (using the Goldwasser-Sipser lower-bound protocol). Let $H \subseteq [k]$ denote the set of queries that the prover claims is heavy. Second, if z_i is not heavy and there exists x such that $C(x) = z_i$, the prover sends x to the verifier. Finally, if z_i is not heavy and z_i is not in the image of C, the prover replies that z_i is not in the image of C.

4. The verifier checks that all the heavy queries are indeed heavy (*i.e.* the proofs of heaviness pass), and if not aborts. The verifier checks that $\frac{|H \cap S|}{|S|}$ is roughly equal to p_{heavy}; if it is too different, then the verifier aborts.

5. Next the verifier checks that for every $i \in [k] \setminus S$, the prover responded with a correct preimage of z_i (which the verifier knows because it sampled preimages along with the queries for $i \in [k] \setminus S$), and if not it aborts.

6. The verifier checks that for every $i \in S$ each preimage x_i that the prover sent is valid by checking $C(x_i) = y_i$ for the corresponding y_i. Let d be the number of queries in $S \setminus H$ that have a (valid) preimage under C. Let $\frac{d}{|S \setminus H|}$ be the verifier's estimate for p_{image}.

The challenge is to prevent the prover from claiming that a query y drawn from D does not have a preimage under C when in fact it does. The idea is that by mixing queries drawn from D among many queries drawn from C, the verifier can "hide" the queries from D among the queries for C. For y drawn from C, the verifier can make sure the prover is behaving honestly since it knows that these queries have preimages, and in fact it can sample the preimage along with y.

This hiding works as long as the queries from D do not occur with too high probability, which is why it needs to exclude the heavy queries from its calculations. For the queries that are not heavy, the prover cannot tell whether they were drawn from C or for D. Since the prover cannot cheat on queries drawn from C and it cannot tell which queries are drawn from C and which from D, the prover has no choice but to answer (almost) all of the queries honestly.

Emulation: in this phase the verifier emulates the execution of the reduction R many times in parallel, using the prover to answer oracle queries. Let $\vec{y}$ denote the vector of queries that a single execution of the reduction queries, and let $\vec{y}_1, \ldots, \vec{y}_\ell$ denote the vectors of queries from the $\ell = \text{poly}(n)$ parallel executions of R that the verifier emulates. The verifier asks the prover to answer these queries along with additional checks as follows:

1. The verifier sends $\vec{y}_1, \ldots, \vec{y}_\ell$ to the prover.

2. The prover responds for each query whether or not it is heavy. Let Q be the set of all queries. Let H denote the set of queries the prover claims are heavy. The prover engages the verifier in a lower-bound protocol to prove that each query in H is heavy. The verifier aborts and rejects if any of these lower-bound protocols fail. The verifier checks that $\frac{|Q \cap H|}{|Q|}$ is roughly p_{heavy}; if it is too far, the verifier aborts.

3. The prover responds for each query whether or not it is in the image, and if so it also includes the preimage. Let I denote the set of queries that are in the image; the verifier checks that for every query in I, the corresponding preimage the prover provides indeed maps to that query. The verifier checks that $\frac{|I \setminus H|}{|Q \setminus H|}$ is roughly p_{image}. If any of these checks fail, the verifier aborts and rejects.

4. The verifier selects a random $i \in [\ell]$ and outputs the result of emulating the reduction using $\vec{y}_i$. It responds to the reduction's queries as follows: if the query is heavy, then the verifier responds that the query is not in the image, and if the query is not heavy, then the verifier responds with the corresponding inverse it obtained from the prover.

The reason this strategy succeeds is because the prover cannot cheat on many of the queries: first, the prover cannot claim a query is in the image when it is not, since there exists no valid preimage it could send to the verifier as a witness. Therefore, the prover can only cheat by saying that a query is not in the image when it actually is, therefore biasing p_{image} to be smaller than it actually is. He cannot bias it by much, since otherwise the verifier would reject. If he cannot bias it by much, then when the verifier picks a random execute to use in the last step, with high probability the verifier will pick an execution where the prover has given completely correct answers with the exception of the heavy queries. But since the reduction should work even if its inverting oracle makes mistakes, the reduction should be able to tolerate mistakes on the heavy queries, of which there cannot be too many. Therefore by the correctness of the reduction the verifier obtains the correct answer with high probability.

General case: to generalize this proof, it is necessary to use more clever protocols to prevent the prover from cheating not only by falsely claiming that queries are outside the image, but also to prevent the prover from adversarially picking the "worst

possible preimage" based on the queries it has seen so far (since in the general case there may be many preimages). This might make the verifier's emulation of the reduction output the incorrect answer, since the reduction assumes that its oracle is not malicious and its responses should not depend on the queries seen so far. [AGGM06] solves this by using a hashing mechanism to force the prover to provide an almost-random preimage, and their technique applies in our setting as well. ■

This concludes the proof of Lemma 5.3.7 and therefore also of Theorem 5.3.4 ■

Remark 5.3.8. With a fixed-auxiliary-input reduction, the above protocol needs only to compute p_{heavy} and p_{image} with respect to the fixed auxiliary input. If the reduction queried multiple auxiliary inputs, then the above protocol would need to compute p_{heavy} and p_{image} with respect to *each* auxiliary input. This is because the inverting oracle promises to successfully invert with high probability for every auxiliary input, and therefore the verifier must be able to prevent the prover from cheating on every auxiliary input.

This becomes problematic if the fixed-auxiliary-input condition is removed, since the auxiliary inputs and inverter queries could be sampled dependently. Namely, the reduction R could sample jointly a circuit C^ω and a query y^ω and ask (C^ω, y^ω) to the inverter. This means that the query distribution of C^ω is the *conditional distribution* on y given that the auxiliary input is C^ω, and this conditional distribution could be unsamplable by efficient algorithms

To see a concrete (albeit contrived) example of why the technique of [BT06, AGGM06] breaks down, consider a reduction R that queries its inverting oracle as follows: first sample a uniformly random y, then apply a one-way function f to get $C = f(y)$, interpret C as a circuit, and query (C, y). In order to compute p_{heavy} and p_{image}, the verifier would need to sample more y from the query distribution corresponding to

auxiliary input C. But this conditional distribution is $f^{-1}(C)$, which by assumption is hard to sample because f is one-way. Therefore the above proof breaks down when we remove the fixed-auxiliary-input condition.

5.4 Summary

In summary, we have shown that the hardness of learning lies between **ZK** and **NP**. Furthermore, if we consider standard techniques, it is unlikely that we can prove that the hardness of learning is equivalent to either the hardness of **NP** or the non-triviality of **ZK**. These relationships are depicted in Figure 5.4.

Open questions: what can one say about (not necessarily strongly) black-box reductions from **NP**-hardness to learning that have super-constant rounds of adaptivity? Can one use highly adaptive or non-black-box techniques to prove that $\mathbf{P} \neq \mathbf{NP}$ implies learning is hard?

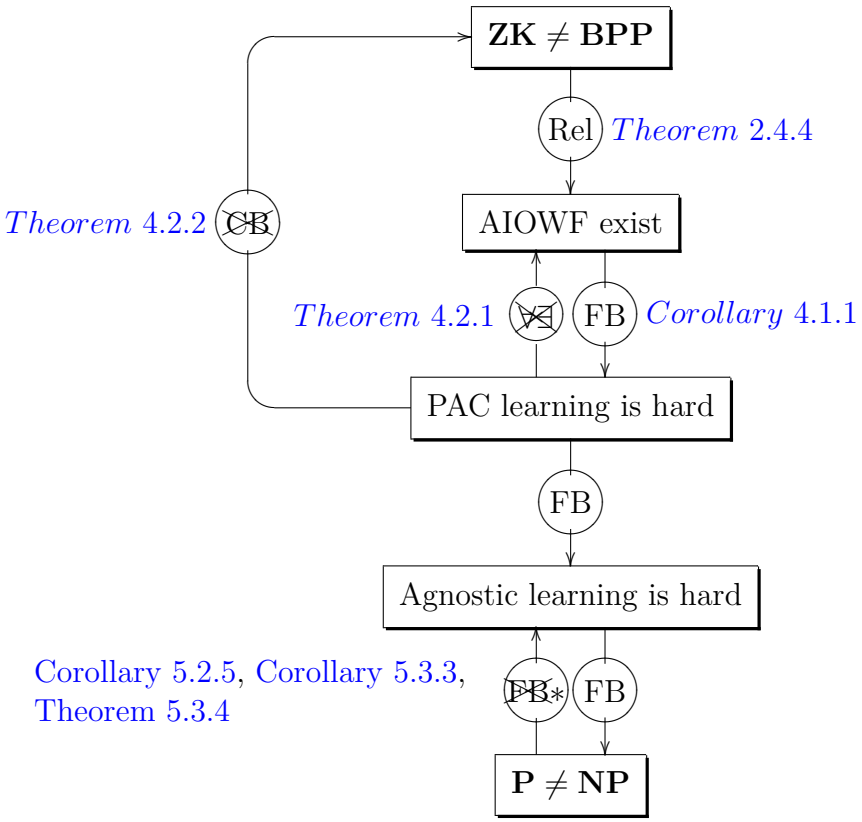

Figure 5.1: Summary of results about learning

Chapter 6

Lower-bounds for failure localization

The Internet is an indispensable part of our society, and yet its basic foundations remain vulnerable to attack. Secure routing protocols seek to remedy this by not only providing guarantees on the correct setup of paths from sender to receiver through a network (*e.g.* secure BGP [KLS00]), but also by verifying that data packets are actually delivered correctly along these paths. Packet delivery is surprisingly susceptible to simple attacks; in the current Internet, packets are typically sent along a single path from sender to receiver, and so a malicious node along the data path can easily drop or modify packets before they reach their destination. To detect and respond to such attacks, the networking community has recently been studying monitoring and measurement protocols that are used to obtain information about packet loss events on a data path (*e.g.* [CK06, DG01, MSWA03, SRS+04, PS03, AHNRR02, AKWK04, MCMS05, AMCS04]). The motivation for such protocols is twofold. First, they provide the sender with information that he can use during path setup to select a single, high-performance path to the receiver from the multiple available paths

through the network [HR08]. Second, since Internet service is a contractual business, where senders pay nodes along the data path to carry their packets, information from Internet measurement protocols is highly valuable for enforcing contractual obligations between nodes. In fact, Laskowski and Chuang [LC06] recently argued that this information is not only valuable, but also *necessary* to counter the Internet industry's growing trend towards degraded path performance. Note that if Internet measurement protocols are used to enforce contractual obligations, nodes may have an economic incentive to bias the information obtained from these protocols.

In this chapter we provide a rigorous cryptographic examination of *secure* monitoring protocols that are robust even in the presence of malicious nodes on the data path. In particular, we study techniques that allow a sender to *localize* the specific links along the data path where packets were dropped or modified.

6.1 Overview of results

We make the following contributions to the study of secure failure-localization path-quality monitoring protocols (in the rest of the chapter we call these simply *failure localization* or FL protocols). Throughout the chapter, we use the word "packet" to denote data that the sender wishes to transmit, and "message" to refer to both data packets and FL-protocol-related messages.

Definition. In Section 6.2, we give the first formal definition of security for failure localization protocols. An important feature of our definition is that it accounts for the fact that messages can be dropped in the Internet for benign reasons like congestion.

Lower bounds. We prove lower bounds that highlight the kind of care necessary

to design protocols that are simultaneously secure against adversarial behaviour in an environment where benign congestion occurs. Specifically, in Section 6.3 we prove that all nodes on the path must participate in the failure localization protocol by sharing keys and performing cryptographic operations. We do so in three steps:

1. Proving that every secure FL protocol requires a key infrastructure, or more precisely, that intermediate nodes and Alice and Bob must all share some secret information between each other.

2. Proving that a one-way function can be constructed from any secure FL protocol.

3. Giving evidence that any practical secure FL protocol must use these keys in a cryptographic way at *every node* (*e.g.* , it does not suffice to use the secret information with some simple, non-cryptographic, hash functions as in [DG01]). We show that in every black-box construction of such a protocol from a random oracle, where at most $O(\log n)$ protocol messages are added per packet, then every intermediate node must query the random oracle. We note that known protocols designed for Internet routers currently avoid using public-key operations, non-black-box constructions, or adding more than a constant number of protocol messages per packet.

Implications of our results. Our lower bounds raise questions about the practicality of deploying FL protocols. In small highly-secure networks or for certain classes of traffic, the high key-management and cryptographic overhead required for FL protocols may be tolerable. However, FL protocols may be impractical for widespread deployment in the Internet; first because intermediate nodes are owned by competing business entities that may have little incentive to set up a key infrastructure and agree

on cryptographic protocols, and second because cryptographic computations are expensive in the core of the Internet, where packets must be processed at extremely high speeds (about 2 ns per packet). Thus, our work can be seen as a motivation for finding security functionalities for the Internet that are more practical than failure localization.

Impact of our methodology. We believe that our methodology can be more widely applied to problems in security. Although efficiency of black-box constructions has been widely studied in theoretical cryptography (*e.g.* [KST99, GT00, GGK03, HHRS07]), we believe that it can be leveraged even more to deepen our understanding of problems in areas of cryptography or security closer to practice. That is, in order to understand the extent to which a particular security requirement is achievable, one should first define a formal model of security and then see to what extent *black-box techniques* can be used to build secure protocols realizing the security requirement. Black-box techniques roughly capture the limit of "practical" cryptography, and therefore if one can prove lower-bounds on the efficiency of black-box constructions realizing the security requirement (as in the case of our work on failure localization), this would imply that the security requirement is too stringent and should be relaxed or modified so that the black-box lower-bounds may be circumvented.

6.1.1 Related work

This work is related to [GXB+08], which gives formal definitions and lower bounds for the simpler task of *path-quality monitoring* (PQM). In a PQM protocol the sender only wishes to *detect* if a failure occurred, rather than localize the specific faulty link along the path, and our lower bounds do not apply in this simpler setting. We construct

highly efficient schemes that are secure for this simpler problem, where intermediate nodes do not need to actively participate in the protocol. This suggests that path-quality monitoring is a more reasonable security goal than failure localization.

In addition to the FL protocols from the networking literature [AKWK04, AHNRR02, PS03, MCMS05, AMCS04, WBA+07], our work is also related to the work on secure message transmission (SMT) begun by Dolev, Dwork, Waart, and Yung in [DDWY93]. In SMT, a sender and receiver are connected by a multiple parallel wires, any of which can be corrupted by an adversary. Here, we consider a single path with a series of nodes that can be corrupted by an adversary, instead of multiple parallel paths. Furthermore, while multiple parallel paths allow SMT protocols to *prevent* failures, in our single path setting, an adversarial intermediate node can always block the communication between sender and receiver. As such, here we only consider techniques for *detecting and localizing* failures.

6.1.2 Historical notes

The work in this chapter was first presented in the paper [BGX08].

6.2 Definition of Secure Failure Localization

In a failure localization (FL) protocol, a sender Alice wants to know whether the packets she sends to receiver Bob arrive unmodified, and if not, to find the link along the path where the failure occurred (see Figure 6.1). We say a *failure* or *fault* occurs when a data packet that was sent by Alice fails to arrive unmodified at Bob. Following the literature, we assume that Alice knows the identities of all the nodes of the data path. We work in the setting where all traffic travels on symmetric paths

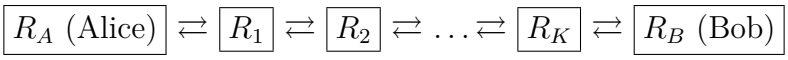

Figure 6.1: A path from Alice to Bob via K intermediate nodes.

(*i.e.* intermediate nodes have bi-directional communication links with their neighbors, and messages that sender Alice sends to receiver Bob traverse the same path as the messages that Bob sends back to Alice). We say that messages travelling towards Alice are going *upstream*, and messages travelling towards Bob are going *downstream*. An adversary Eve can occupy any set of nodes on the path between Alice and Bob, and can add, drop, or modify messages sent on the links adjacent to any of the nodes she controls. She can also use timing information to attack the protocol.

Localizing links, not nodes. It is well known that an FL protocol can only pinpoint a *link* where a failure occurred, rather than the *node* responsible for the failure. To see why, refer to Figure 6.1, and suppose that (a) Eve controlling node R_2 becomes unresponsive by ignoring all the messages she receives from R_1. Now suppose that (b) Eve controls node R_1 and pretends that R_2 is unresponsive by dropping all communication to and from R_2. Because cases (a) and (b) are completely indistinguishable from Alice's point of view, at best Alice can localize the failure to link $(1, 2)$.

Congestion. Congestion-related packet loss is widespread on the current Internet, caused by protocols like TCP [Jac88] that naturally drive the network into a state of congestion. Our definition accounts for congestion by assuming links can drop each message independently with some probability. One could come up with other models for congestion (*e.g.* allowing Eve to specify the distribution of congestion-related packet loss), however, we use independent drops for the sake of simplicity. Furthermore, assuming that congestion is not controlled by the adversary only strengthens our negative results and makes our model more realistic.

6.2.1 Security definition

Let n be the security parameter. A failure localization protocol consists of an efficient initialization algorithm Init taking n uniformly random bits and generating keys for each node, and efficient node algorithms Alice, Bob, $R_1, \ldots, R_K$ which take in a key and communicate with each other as in Figure 6.1. The Alice algorithm takes in a packet that she wants to send to Bob. If communication is successful, then the Bob algorithm outputs the packet that Alice sent. Our security definitions are game-based:

Definition 6.2.1 (Security game for FL)**.** The game begins when Eve chooses a subset of nodes $E \subseteq \{1, \ldots, K\}$ that she will occupy for the duration of the game. The Init algorithm is then used to generate keys for each node, and Eve is given the keys for the nodes $i \in E$ that she controls. We define an oracle Source that generates data packets d for the Alice algorithm to send. We allow Eve to choose the packets that the Source oracle generates, subject to the condition that she may not choose the same packet more than once during the game. We make this assumption because there is natural entropy in packet contents, due to TCP sequence numbers and IP ID fields [DG01]. To enforce this assumption in practice, protocol messages can be timestamped with with an expiry time, such that with high probability (over the entropy in the packet contents), no repeated packets are sent for the duration of the time interval for which the protocol messages are valid. We allow Eve to add, drop, or modify any of the messages sent on the links adjacent to the nodes she occupies. We include congestion in our model by requiring that, for each message sent on each link on the path, the link *goes down* or drops the message with some constant probability $\rho > 0$. Notice that this means that a failure can happen at links not adjacent to a node occupied by Eve.

We introduce the notion of time into our model by assuming that the game proceeds

in discrete timesteps; in each timestep, a node can take in an input and produce an output, and each link can transmit a single message. (Thus, each timestep represents an event occurring on the network.) Because it is expensive to have securely synchronized clocks in a distributed system like the Internet we do *not* allow the honest algorithms to take timing information as an input. However, to model timing attacks, we assume that Eve knows which timestep that the game is in.

Our security definition uses the the game defined in Definition 6.2.1:

Definition 6.2.2 (Security for FL)**.** In the security game, Eve gets to interact with the Source oracle and the "honest" node algorithms as in Definition 6.2.1, until she decides to stop. For each packet sent, Alice must output either $\surd$ (*i.e.* not raise an alarm) or a link ℓ (*i.e.* raise an alarm and localize a failure to ℓ). We assume that the game is *sequential:* Alice must output a decision for each data packet before starting to transmit the next data packet (see remarks below). We say that an FL protocol is *secure* if the following hold:

1. (*Secure localization*). For every packet d sent by the Source oracle that is not successfully output by Bob, then Alice outputs a link ℓ such that either (a) link ℓ is adjacent to a node occupied by Eve, or (b) link ℓ went down due to congestion for one of the messages (including FL protocol messages) associated with sending packet d from Alice to Bob.

2. (*No false positives*). For every packet d sent by the Source oracle that is successfully output by Bob, for which there was no congestion, and for which Eve does not deviate from the protocol, Alice outputs $\surd$.

We now discuss some properties of our security definition.

Benign and malicious failures. Our security definitions require Alice to accurately localize failures, but these failures may be caused by Eve, or may be the result of *benign causes*, such as congestion. We do not require Alice to distinguish between benign or malicious (*i.e.* due to Eve) failures, because Eve can always drop packets in a way that "looks like" congestion.

Sequential games. For simplicity, in our security game we required Alice to make FL decisions before she sends a new data packet. This is to capture the fact that such protocols should provide "real-time" information about the quality of the paths she uses, and so we did not allow Alice to make decisions only after sending many packets. We emphasize that the sequential assumption does *not* prevent Alice from keeping state and using information from *past* packets in order to make FL decisions.

Movements of the adversary. Our model does not allow Eve to move from node to node in a single security game. This assumption makes sense when Eve models a Internet service provider that tries, for business reasons, to bias the results of FL protocol. Furthermore, when Eve is an external attacker or virus that compromises a router, "leaving" a router means that the legitimate owner of the router removed the attacker from the router, *e.g.* by refreshing its keys. We model this key refresh process as a re-start of the security game. Furthermore, in practice "movements" to a new router happen infrequently, since an external attacker typically needs a different strategy each time it compromises a router owned by a different business entity.

Generalizations. All our results generalize to the setting where congestion rates, false alarm thresholds, and detection thresholds are different per link; we set them all equal here for simplicity. Our negative results also hold for the weaker adversary model where Eve can occupy only one node and the Source oracle generates independent (efficiently-samplable) packets from a distribution that is *not* controlled

by Eve. In [BGX08] we also study *statistical* notions of security, where Alice is not required to discover the location of individual failures but should discover if an adversary is disrupting a particular link consistently over many packets.

6.3 Security requires keys required at each node

We now argue that in any secure FL scheme Alice requires shared keys with Bob and the intermediate nodes, and Alice, Bob and each intermediate node must perform cryptographic operations. We only argue for intermediate nodes $R_2, \ldots, R_K$; R_1 is a border case which requires neither keys nor crypto because we assume Alice is always honest.

Since FL provides strictly stronger security guarantees than path-quality monitoring, it follows from the results in [GXB+08] that in any secure FL protocol, Alice and Bob must have shared keys. We also have the following theorem that proves that in any secure FL protocol, *each* intermediate node must share keys with some Alice:

Theorem 6.3.1. *Suppose* Init *generates some auxiliary information* aux_i *for each node* R_i *for* $i = 1, ..., K, \mathsf{Alice}, \mathsf{Bob}$. *A FL protocol cannot be secure if there is any node* $i \in \{2, \ldots, K\}$ *such that* $(\mathsf{aux}_{\mathsf{Alice}}, \mathsf{aux}_1, \ldots, \mathsf{aux}_{i-1})$ *and* aux_i *are independent.*

Proof. Suppose R_i has aux_i that is independent of $(\mathsf{aux}_{\mathsf{Alice}}, \ldots, \mathsf{aux}_{i-1})$. Then, the following two cases are indistinguishable from Alice's view: (a) Node $R_{i+1}$is malicious and blocks communication on link $(i, i+1)$, and (b) Eve occupies node R_{i-1}, and drops packets while simulating case (a) by picking an independent aux'_i and running $R_i(\mathsf{aux}'_i)$ while pretending as if $(i, i+1)$ is down. These two cases are indistinguishable because aux_i is independent of $(\mathsf{aux}_{\mathsf{Alice}}, \ldots, \mathsf{aux}_{i-1})$, and so Alice will localize the failure to the same link in both case (a) and (b). But this breaks security, since R_{i+1}, R_{i-1} do

not share a common link. ■

6.4 Security requires crypto at each node

[GXB+08] proves the following:

Theorem 6.4.1 ([GXB+08])**.** *The existence of a secure PQM protocol implies the existence of an infinitely-often one-way function (i.o.-OWF).*

Since one-way functions are equivalent to many cryptographic primitives (in the sense that these primitives exist if and only if one-way functions exist [IL89]), this result can be interpreted to mean that nodes participating in any secure PQM protocol must perform cryptographic computations. Since FL gives a strictly stronger security guarantee than PQM, we also have that in any FL protocol, some node on the data path must perform cryptography. However, Theorem 6.4.1 only implies that the *entire system* performs cryptography. We want to prove that any secure FL protocol requires *each intermediate node* $R_1, \ldots, R_K$ to perform cryptography. Because it is not clear even how to formalize this in full generality, we instead apply the methodology of Impagliazzo and Rudich [IR89] to do this for *black-box* constructions of FL protocols from a random oracle RO. We model "performing cryptography" as querying the random oracle, and show that in such a secure FL protocol *each node* must query the RO. In the language of Definition 1.3.3, we will prove there exists no relativizing reduction that uses OWF to construct a black-box FL protocol where one node never queries the OWF can be broken. We do this by showing that for every such protocol, relative to the oracle $(\mathsf{RO}, \mathbf{PSPACE})$ there exists OWF but the protocol can be broken.

Before stating the theorem and proof, we set up some terminology. We will use the

notion of an *exchange* to denote a data packet and all the FL-protocol-related messages associated with that packet. Because our game is sequential (see Section 6.2), Alice's must decide to localize a link ℓ or output $\surd$ before the next exchange begins. Let $\langle R_{i-1}, R_i \rangle$ denote the distribution of transcripts (*i.e.* all messages sent and received) on link $(i-1, i)$. Because we allow the nodes to keep state, this distribution may depend on what happened in all previous exchanges; therefore $\langle R_{i-1}, R_i \rangle$ denotes the distribution of the next exchange concatenated with the transcript of all previous exchanges that have occurred so far in the security game. We assume without loss of generality that the number of messages per exchange is even, and that odd messages go from R_{i-1} to R_i and even messages go from R_i to R_{i-1}. We let r denote the number of rounds in an exchange, where one round consists of a message from R_{i-1} to R_i and a response from R_i to R_{i-1}. Protocols where the number of messages per exchange grows quickly with n are impractical and so we restrict our attention to "practical" protocols where each exchange contains at most $r = O(\log n)$ rounds.

Theorem 6.4.2. *There is no relativizing reduction that uses OWF to construct a FL protocol such that at least one node R_i for $i \in \{2, \ldots, I\}$ never calls the OWF, and where the maximum number of messages per exchange is $O(\log n)$.*

In fact, we prove something stronger, namely the following:

Theorem 6.4.3. *Fix a fully-black-box reduction that uses a random oracle to build a secure FL protocol such that at least one node R_i for $i \in \{2, \ldots, I\}$ never calls the* RO *and where the maximum number of messages per exchange is $O(\log n)$. Then there exists an efficient algorithm relative to* $(\mathsf{PSPACE}, \mathsf{RO})$ *that breaks the security of the scheme with non-negligible probability over the choice of random oracle* RO *and the internal randomness of the algorithm.*

Theorem 6.4.2 follows from Theorem 6.4.3 by observing that relative to $(\mathsf{RO}, \mathbf{PSPACE})$, one-way functions exist.

Proof overview: We construct an adversary, which we call Eve, that controls node R_{i-1} and whose goal is to impersonate R_i. Eve is allowed oracle access to $(\mathsf{RO}, \mathbf{PSPACE})$. The attack is similar in spirit to the attack in Theorem 6.3.1, but now aux_i is secret, so Eve must first *learn* aux_i. Eve's overall strategy consists of two phases:

1. *Learning to impersonate.* Sitting at R_{i-1}, Eve observes at most t exchanges (t is polynomial in n) on the link $(i-1, i)$, where in each exchange Eve asks Source to transmit a uniformly random data packet. She then uses the learning algorithm of Naor and Rothblum [NR06] to obtain a pair of impersonator algorithms A', B', whose interaction generates a distribution over transcripts for the $t+1$'th exchange. A' impersonates nodes $\mathsf{Alice}, R_1, \ldots, R_{i-1}$ and B' impersonates nodes $R_i, \ldots, R_K, \mathsf{Bob}$.

2. *Dropping and impersonating.* On the $t+1$'th exchange, for each message going from R_{i-1} to R_i, Eve replies with a response she computes herself using a modified version of B'; she does not send any messages to R_i. (See Definition 6.4.6 for what this modified version of B' means).

Now, Eve at R_{i-1} will break security if she manages to use B' to impersonate an *honest* exchange during which link $(i, i+1)$ is down. (This breaks security since link $(i, i+1)$ is not adjacent to R_{i-1}.) The crucial observation is that here, Eve need only impersonate node R_i since R_i is disconnected from R_{i+1}, and that R_i does not "protect" its secret keys by calling the RO. Intuitively, Eve should be able to impersonate R_i since any computations that R_i does are easy to invert relative to $(\mathsf{PSPACE}, \mathsf{RO})$.

To prove the theorem, we shall show that with non-negligible probability $> (10/\rho)^r = 1/\text{poly}(n)$ where $r = O(\log n)$ is the number of rounds in an exchange, the following are 1/100-indistinguishable: (a) Alice's view when link $(i, i+1)$ is down and (b) Alice's view when R_{i-1} drops a packet but impersonates link $(i, i+1)$ being down using B'.

The learning algorithm

Recall (Section 6.2) that Alice is allowed to use information from past exchanges to help her decide how to send messages in new exchanges. Fortunately, the algorithm of Naor and Rothblum [NR06] is specifically designed to deal with this, and guarantees the following:

Lemma 6.4.4 (Based on [NR06])**.** *Relative to a* $(\mathsf{PSPACE}, \mathsf{RO})$*-oracle, there exists an efficient algorithm that observes at most* $t = O(\frac{n}{\varepsilon^4})$ *honest exchanges* $\langle R_{i-1}, R_i \rangle_{1,\ldots,t}$ *and then, with probability* $> 1-\varepsilon$*, outputs efficient impersonator algorithms* $R'_0, \ldots, R'_{K+1}$ *such that an impersonated transcript* $\langle R'_{i-1}, R'_i \rangle_{t+1}$ *(given by simulating the interaction of all the impersonator algorithms) for the exchange* $t+1$ *is distributed* ε*-close in statistical distance to the honest transcript* $\langle R_{i-1}, R_i \rangle_{t+1}$ *for exchange* $t+1$*.*

Let A' denote the algorithm that emulates the interaction of $R'_0, \ldots, R'_{i-1}$ and B' denote the algorithm that emulates the interaction $R'_i, \ldots, R'_{K+1}$ that satisfy the guarantee above. The transcript $\langle A', B' \rangle$ is therefore exactly the transcript $\langle R'_{i-1}, R'_i \rangle$.

To prove the theorem we have to overcome several challenges.

1. Naor-Rothblum algorithm only guarantees that $\langle A', B' \rangle$ and $\langle R_{i-1}, R_i \rangle$ are statistically close, but does *not* guarantee that $\langle A', B' \rangle \circ \mathsf{RO}$ is statistically close to the "honest" transcript $\langle R_{i-1}, R_i \rangle \circ \mathsf{RO}$, *i.e.* if the value of RO is also known.

Fortunately, we will be able to exploit the fact that with probability ρ^r all the messages sent from R_i to R_{i-1} are computed independent of RO. This happens when *congestion* causes link $(i, i+1)$ to go down for the duration of an exchange (so that R_i, who never calls the RO, has to compute all his upstream messages on his own).

2. Eve has no control, or even knowledge, of when congestion causes this event to occur. Indeed, the fact that $\langle A', B' \rangle$ is close to $\langle R_{i-1}, R_i \rangle$ does not guarantee that the same holds if we *condition on congestion occurring* $(i, i+1)$. For this reason, Eve cannot simply use R'_i (instead of $R'_i, ..., R'_K, \mathsf{Bob}'$) to impersonate the honest R_i conditioned on link $(i, i+1)$ being down. Fortunately, we can show that with probability ρ^r, A', B' will generate a "useful" impersonated transcript that is ε/ρ^r-statistically close to the honest transcripts conditioned on the event that link $(i, i+1)$ is down. Eve does not necessarily know *when* she impersonates a useful transcript; she simply has to hope that she is lucky enough for this to happen.

3. Even when Eve is lucky enough to obtain a useful transcript, what we really want to prove is that (a) $\langle R_{i-1}, B' \rangle$ conditioned on getting a useful transcript is statistically close to (b) $\langle R_{i-1}, R_i \rangle$ conditioned on congestion occuring on link $(i, i+1)$. That is, we care not about $\langle A', B' \rangle$ which is the transcript of the interaction of a pair of impersonated algorithms, but rather $\langle R_{i-1}, B' \rangle$ which is the interaction of *honest* algorithms $R_0, \ldots, R_{i-1}$ and *impersonated* algorithms $R'_i, \ldots, R'_{K+1}$. We prove that, with probability at least $(\rho/2)^r$, the impersonator algorithm B' interacting with honest $R_0, ...R_{i-1}$ still generates a useful transcript such that the statistical distance between (a) and (b) is at most $1/100$.

The Interaction Lemma

We address these challenges in the next lemma. We state a general version of the lemma here, for which we first need a few definitions. Let A, B and A', B' be two (different) pairs of algorithms such that the statistical difference between the transcripts $\langle A, B\rangle$ and $\langle A', B'\rangle$ is bounded by ε. The parties are randomized, *i.e.* A can use random coins ω_A, B can use random coins ω_B and furthermore we allow A, B to use shared random coins, which we call ω_{AB}. The same holds for A', B'.

We need to define A's point of view in an interaction. Even though A is "supposed" to talk to B, this view can be defined for any partner, not just B.

Definition 6.4.5 (View of A, view_A)**.** $\mathsf{view}_A(A, C)$ is a distribution over $(\tau, \omega_A, \omega_{AB})$ where τ is a transcript of the interaction between A, C and ω_A, ω_{AB} are the private and shared random coins of A. $\mathsf{view}_A(A, C)$ is obtained by first sampling uniform coins ω_A, ω_{AB}, and then generating τ by interacting with C using these random coins. When $C = B$, we allow B to use the shared random coins ω_{AB}, otherwise we assume C generates its messages independently of ω_A, ω_{AB}.

In the sequel, we will use a modified B' that samples its messages by "reverse sampling".

Definition 6.4.6 (Reverse sampling for B')**.** Given a prefix of messages $\tau_{2j-1} = (m_1, \ldots, m_{2j-1})$, B' samples the $2j$'th message by sampling from the conditional distribution $\langle A', B'\rangle_{2j} \mid \langle A', B'\rangle^{2j-1} = \tau_{2j-1}$. Here, $\langle A', B'\rangle_{2j}$ denotes the $2j$'th message of $\langle A', B'\rangle$, while $\langle A', B'\rangle^{2j-1}$ denotes the concatenation of the first $2j-1$ messages of $\langle A', B'\rangle$.

In general this "reverse sampling" is not efficient, but it is efficient relative to a PSPACE oracle. Furthermore, it is clear from the definition that if one generates

the prefix τ_{2j-1} according to $\langle A', B' \rangle^{2j-1}$, then the transcript $\tau_{2j-1} \circ m_{2j}$ is generated from the same distribution $\langle A', B' \rangle^{2j}$ regardless of whether we think of m_{2j} as being computed in the normal way by B''s next message function, or as being computed using the reverse sampling procedure.

We are finally ready for the statement of the Interaction Lemma.

Lemma 6.4.7 (Interaction Lemma)**.** *Let $(A, B), (A', B')$ be two pairs of algorithms that interact for at most r rounds to produce a transcript. Suppose that*

$$\Delta(\langle A, B \rangle, \langle A', B' \rangle) \leq (\rho/10)^{4r}$$

Suppose there exist events $E_1, \ldots, E_r$ over the internal randomness of A, B such that both the following hold:

1. *$\forall j \in [r]$, conditioned on E_j, the first j messages from B to A are independent of A's coins ω_A, ω_{AB}.*
2. *$\Pr[E_j \mid E_{j-1}] \geq \rho$.*

Then there exist $\eta \geq (\rho/2)^r$, and distributions over the transcripts Y, Z such that the distribution $\mathsf{view}_A(A, B')$ is a convex combination $\eta Y + (1-\eta)Z$ of two distributions Y and Z such that Y satisfies:

$$\Delta(Y, (\mathsf{view}_A(A, B) \mid E_r)) \leq 1/100$$

Proof of Theorem 6.4.3

Proof of Theorem 6.4.3. Lemma 6.4.7 tells us that, with probability η, $\langle A, B' \rangle$ will generate a "useful" transcript Y that is 1/100-statistically close to the honest transcript $\langle A, B \rangle$ conditioned on event E_r occurring. (Z is the "not useful" transcript

that is generated with probability $1-\eta$.) We can now apply Lemma 6.4.7 to our setting:

- A to be honest algorithms $R_0, R_1, \ldots, R_{i-1}$.
- B to be honest algorithms $R_i, \ldots, R_{K+1}$.
- A' to be the impersonator algorithms $R'_0, \ldots, R'_{i-1}$ given by Lemma 6.4.7.
- B' to be the impersonator algorithms $R'_i \ldots, R'_{K+1}$ given by Lemma 6.4.7.
- E_j to be the event that link $(i, i+1)$ is congested for the messages $1, \ldots, j$ that are sent downstream from B to A. (Then, E_r is the event that link $(i, i+1)$ drops all messages from B to A in the exchange).

Now, notice that since R_i does not query the random oracle, conditioned on E_j the first j messages of B are independent of A because they are computed by R_i only. Next, note that $\Pr[E_j \mid E_{j-1}] = \rho$ because each message is lost to congestion at random independently.

To combine everything, Eve sets $\varepsilon = (\rho/10)^{4r}$ and applies the learning algorithm of Lemma 6.4.4 to obtain with probability at least $\geq (1-\varepsilon)$ a pair of algorithms A', B' that is ε-close to $\langle A, B\rangle$ (notice that Eve is efficient since $\varepsilon = 1/\text{poly}(n)$).

Conditioned on getting such A', B', then Eve uses B' to interact with A by generating a transcript according to the distribution $\langle A, B'\rangle$ using reverse sampling, as in Definition 6.4.6. This reverse sampling is efficient since Eve has a **PSPACE** oracle.

Applying the Interaction Lemma (Lemma 6.4.7) we get that with probability $\eta = (\rho/2)^r = 1/\text{poly}(n)$, Eve is lucky enough to generate a useful transcript such that the view of Alice when Eve drops a packet at R_{i-1} and impersonates using $R'_i, \ldots, R'_{K+1}$ is 1/100-indistinguishable from the situation where link $(i, i+1)$ is completely down

for the duration of an exchange. Alice localizes the same link in both cases with probability 99/100 because her output depends solely on her view, so this breaks security since link $(i, i+1)$ is not adjacent to Eve at R_{i-1}. ■

6.4.1 Proof of Lemma 6.4.4

First a word about random oracles, which we treat as a function $\mathsf{RO} : \{0,1\}^* \to \{0,1\}$. We look at the RO using the "lazy evaluation" methodology: points in RO are not fixed until they have been queried. When an efficient algorithm executes with a random oracle, it can make only an efficient number of queries. This means that RO can be viewed as a polynomially long string representing the responses to the algorithm, rather than as an infinitely large function, and replacing RO by a different string RO' (of equal length) amounts to replacing the real random oracle with a "fake random oracle". Thus, in the following, when we say that an oracle outputs a fake random oracle consistent with the output h of an algorithm A, we mean it outputs a string $\mathsf{RO}' \in \{0,1\}^{\mathrm{poly}(n)}$ such that running A with the responses encoded in RO' generates h.

We apply Naor and Rothblum's [NR06] learning algorithm for *adaptively changing distributions (ACD)*. The ACD we work with is defined as a pair (Init, D) of random processes, where Init is a key generation process that takes a uniform string s and generates secrets $\overrightarrow{\mathsf{aux}} = \mathsf{Init}(s)$, and D is a process that takes the initial state $\overrightarrow{\mathsf{aux}}$ and a history H_{i-1} and uses them to generate a sample τ_i of an exchange. The history H_{i-1} consists of tuples (r_j, τ_j) for all $j \leq i-1$, where r_j was the random string used to generate the transcript τ_j. Notice that the τ_j are the outputs of the ACD, while the initial state s and the r_j remain secret.

Theorem 6.4.8 ([NR06])**.** *There exists a* PSPACE *algorithm that, for any ACD*

(Init, D), *observes at most* $t = O(n/\varepsilon^4)$ *samples from* D *and generates with probability* $> 1-\varepsilon$ *a fake secret state* $\overrightarrow{\mathsf{aux}}'$ *and fake history* H'_t *such that simulating* D *with* $\overrightarrow{\mathsf{aux}}', H'_t$ *generates a sample* tau'_{t+1} *that is distributed* ε*-statistically close to an honest sample generated by* D *using* $\overrightarrow{\mathsf{aux}}, H_t$.

Proof of Lemma 6.4.4. Apply Theorem 6.4.8 where the Init function is our key generation function, and D is the algorithm that simulates the interaction of all algorithms $R_0, \ldots, R_{K+1}$ given a uniformly random data packet to be sent, including simulating all the congestion along links between the nodes, and outputs the transcript along link $(i-1, i)$. To generate the transcript of the i'th exchange, D takes input $\overrightarrow{\mathsf{aux}}, H_{i-1}, \mathsf{RO}_i, r_i$ where RO_i are responses to new queries to the random oracle that D makes in generating the transcript, r_i is the fresh internal randomness used to generate the $i+1$'th transcript, and $H_{i-1} = (\tau_j, \mathsf{RO}_j, r_j)_{j \le i-1}$ is a history of previous transcripts, responses of the random oracle, and internal randomness. Notice that because D simulates *all* the nodes, there is *no distinction* between how the learning algorithm treats RO_i and r_i.

After observing $t = O(n/\varepsilon^4)$ exchanges, using the learning algorithm of Theorem 6.4.8, we get with probability $> 1 - \varepsilon$ fake secrets $\overrightarrow{\mathsf{aux}}', H'_t$ consistent with the transcripts and such that generating the $t+1$'th transcript using the fake secrets is ε-close to generating the $t+1$'th transcript using the honest secrets. Set R'_i to be R_i but with the secrets in $\overrightarrow{\mathsf{aux}}', H'_t$ hardwired into the algorithm.

Efficiency is clear because we allow a PSPACE oracle and because the number of samples is $O(n/\varepsilon^4)$. ■

6.4.2 Proof of Lemma 6.4.7

We will usually let τ refer to a fixed transcript, and σ to refer to distributions over transcripts. Define $\sigma_j = \langle A, B\rangle^j$, the random variable for the first j messages in the partial transcript of $\langle A, B\rangle$. Similarly define $\sigma'_j = \langle A', B'\rangle^j$ and $\sigma^{\mathsf{alt}}_j = \langle A, B'\rangle^j$. We will decompose A, B, A', B' into next-message functions A_j, B_j, A'_j, B'_j for $1 \le j \le r$, where we assume that the parties alternate turns communicating, and $2r$ is the maximum number of messages transmitted. Recall from Definition 6.4.6 that the next message function $B'_j(\tau)$ on input τ_{2j-1}, which is a prefix of $2j-1$ messages, is defined by "reverse sampling" the distribution $(\langle A', B'\rangle_{2j} \mid \langle A', B'\rangle^{2j-1} = \tau)$.

Understanding A's view

Definition 6.4.9 (Conditional and alternating views)**.** We define two different views of a transcript that are related to view_A, which we call $\mathsf{condview}_A$ and $\mathsf{altview}_A$.

1. *Conditional view*: $\mathsf{condview}_A(\tau_j)$ for a partial transcript τ_j of the first j messages is the distribution over strings $(\tau, \omega_A, \omega_{AB})$ obtained by sampling uniformly random $\omega_A, \omega_{AB}, \omega_B$ such that when A, B interact using these coins, they produce $\langle A, B\rangle^j = \tau_j$. If there are no such $\omega_A, \omega_{AB}, \omega_B$, set $\omega_A = \omega_{AB} = \perp$.

2. *Alternating view*: $\mathsf{condview}_A(\tau_j)$ for a partial transcript $\tau_j = (m_1, \ldots, m_j)$ of the first j messages is the distribution over strings $(\tau, \omega_A, \omega_{AB})$ obtained by sampling uniformly random ω_A, ω_{AB} such that when using these random coins and the responses $m_2, m_4, \ldots$ contained in τ_j, A produces $m_1, m_3, \ldots$ consistent with τ_j. If there are no such ω_A, ω_{AB}, set $\omega_A = \omega_{AB} = \perp$.

Let $\mathsf{condview}_A(A, C) = \mathsf{condview}_A(\langle A, C\rangle)$ and $\mathsf{altview}_A(A, C) = \mathsf{altview}_A(\langle A, C\rangle)$. We next observe some properties about $\mathsf{view}_A, \mathsf{condview}_A, \mathsf{altview}_A$.

Proposition 6.4.10. *Properties of* condview, altview, view*:*

1. $\mathsf{condview}_A(A, B) = \mathsf{view}_A(A, B)$
2. $\mathsf{altview}_A(A, B') = \mathsf{view}_A(A, B')$
3. *Conditioned on* E_j, $\mathsf{condview}_A(\langle A, B\rangle^{2j}) = \mathsf{altview}_A(\langle A, B\rangle^{2j})$.

Proof. 1. This holds because sampling $\omega_A, \omega_{AB}, \omega_B$ and then generating the transcript is identical to first sampling a transcript $\langle A, B\rangle$ and then generating random coins consistent with the transcript.

2. This holds because B' is independent of A, therefore the transcript generated by sampling ω_A, ω_{AB} and then having A use these random coins to interact with B' is identical to first generating the transcript $\langle A, B'\rangle$ and then reverse sampling consistent coins ω_A, ω_{AB}.

3. Conditioned on E_j, B's messages are independent of A's, and therefore reverse sampling (ω_A, ω_{AB}) consistent with A's messages in the transcript is identical to sampling consistent $(\omega_A, \omega_{AB}, \omega_B)$ consistent with the transcript, then outputting (ω_A, ω_{AB}).

■

We will prove the following by induction:

Claim 6.4.11. *Set* $\varepsilon = (\rho/10)^{4r}$*, and suppose for each* i*,* $0 \le i \le r$*, there exist* $\eta_i \ge \rho/2$ *and random variables* Y_i, Z_i *such that* $\sigma_{2i}^{\mathsf{alt}} = \prod_{j=1}^{i} \eta_j Y_i + (1 - \prod_{j=1}^{i} \eta_j) Z_i$ *and* $\Delta((\mathsf{condview}_A(\sigma_{2i}) \mid E_i), \mathsf{altview}_A(Y_i)) \le \delta_i$*, where* $\delta_i = \sqrt{\varepsilon}(10/\rho)^i$.

Proof of Lemma 6.4.7

Proof of Lemma 6.4.7. Apply this claim for the case of $\sigma_{2r} = \langle A, B \rangle$ and $\sigma_{2r}^{\mathsf{alt}} = \langle A, B' \rangle$ to obtain $Y_r, Z_r, \eta = \prod_{j=1}^{r} \eta_j \geq (\rho/2)^r$ such that both the following hold:

$$\sigma_{2r}^{\mathsf{alt}} = \eta Y_r + (1-\eta) Z_r \tag{6.4.1}$$

$$\Delta(\mathsf{altview}_A(Y_r), (\mathsf{condview}_A(\sigma_{2r}) \mid E_r)) \leq \sqrt{\varepsilon}(10/\rho)^r < 1/100 \tag{6.4.2}$$

By setting $Y = \mathsf{altview}_A(Y_r), Z = \mathsf{altview}_A(Z_r)$ and using the observation from Proposition 6.4.10 (Item 2) that $\mathsf{altview}_A(\sigma_{2r}^{\mathsf{alt}}) = \mathsf{altview}_A(A, B') = \mathsf{view}_A(A, B')$, we get from Equation 6.4.1 the convex decomposition $\mathsf{view}_A(A, B') = \eta Y + (1-\eta) Z$. Finally, by using the observation from Proposition 6.4.10 (Item 1) that conditioned on E_r it holds that $\mathsf{condview}_A(\sigma_{2r}) = \mathsf{condview}_A(A, B) = \mathsf{view}_A(A, B)$, we obtain from Inequality 6.4.2 that

$$\Delta(Y, (\mathsf{view}_A(A, B) \mid E_r)) < 1/100$$

This completes the proof of Lemma 6.4.7 modulo the proof of the Claim 6.4.11 to which we now proceed. ■

Proof of Claim 6.4.11

We use the following technical lemmas. The first lemma says that if X, X' are statistically close and X can be decomposed as $X = \eta Y + (1-\eta) Z$, then X' can be decomposed similarly such that corresponding parts of X and X''s decomposition are statistically close.

Lemma 6.4.12. *For any random variables X, Y, Z, X' satisfying $X = \eta Y + (1-\eta) Z$ and $\Delta(X, X') \leq \varepsilon$, there exists random variables Y', Z' and $\eta' \in [\eta \pm \varepsilon]$ such that $X' = \eta' Y' + (1-\eta') Z'$ and $\Delta(Y, Y') \leq \frac{3\varepsilon}{2\eta}$.*

Proof. Define a randomized process F acting on the support of X, where for each $x \in \text{supp}(X)$, $F(x) = 1$ with probability $p(x) = \frac{\eta \Pr[Y=x]}{\Pr[X=x]}$ and $F(x) = 0$ with probability $1 - p(x)$, and say $F(x) = 0$ for all $x \notin \text{supp}(X)$. We can check that

$$\Pr[F(X) = 1] = \mathbb{E}[F(X)] = \sum_{x \in \text{supp}(X)} \Pr[X = x] \frac{\eta \Pr[Y=x]}{\Pr[X=x]} = \sum_{x \in \text{supp}(X)} \eta \Pr[Y = x] = \eta$$

and similarly $F(X) = 0$ with probability $1 - \eta$. Furthermore, we claim that $Y = (X \mid F(X) = 1)$ since for every x,

$$\Pr[X = x \mid F(X) = 1] = \frac{\Pr[F(X)=1 \wedge X=x]}{\Pr[F(X)=1]} = \frac{\Pr[F(x)=1] \Pr[X=x]}{\eta} = \Pr[Y = x]$$

and similarly $Z = (X \mid F(X) = 0)$.

Since $\Delta(X, X') \leq \varepsilon$, this means that $\Pr[F(X') = 1] = \eta' \in [\eta \pm \varepsilon]$, and also $\Delta((F(X), X), (F(X'), X')) \leq \varepsilon$. Define $Y' = (X' \mid F(X') = 1)$ and $Z' = (X' \mid F(X') = 0)$. We may derive:

$$\begin{aligned} \varepsilon &\geq \Delta((F(X), X), (F(X'), X')) \\ &= \Delta(\eta(1, Y) + (1 - \eta)(0, Z), \eta'(1, Y') + (1 - \eta')(0, Z')) \end{aligned}$$

Viewing the random variables as the characteristic vectors of their distributions, and using the ℓ_1 formulation of statistical distance, we have:

$$= \tfrac{1}{2} \|\eta(1, Y) + (1 - \eta)(0, Z) - \eta'(1, Y') - (1 - \eta')(0, Z')\|_1$$

Since coordinates of the form $(1, Y)$ are disjoint from coordinates of the form $(0, Z)$, we have the equality:

$$\begin{aligned} &= \tfrac{1}{2} \|\eta(1, Y) - \eta'(1, Y')\|_1 + \tfrac{1}{2} \|(1 - \eta)(0, Z) - (1 - \eta')(0, Z')\|_1 \\ &\geq \tfrac{1}{2} \|\eta(1, Y) - \eta'(1, Y')\|_1 \\ &= \tfrac{1}{2} \|\eta Y - \eta Y' + (\eta' - \eta) Y'\|_1 \\ &\geq \tfrac{1}{2} \eta \|Y - Y'\|_1 - \tfrac{1}{2} |\eta' - \eta| \\ &\geq \eta \Delta(Y, Y') - \varepsilon/2 \end{aligned}$$

which, rearranged, gives us that $\Delta(Y, Y') \leq \frac{3\varepsilon}{2\eta}$. ■

For notational convenience, we will let both XY and (X, Y) denote the random variable that takes a sample from the random variable X concatenated with a sample from Y. We will also let $Y(x)$ denote the conditional distribution $Y \mid X = x$ and likewise $Y'(x)$ denotes $Y' \mid X' = x$. The following lemma says that if two pairs of variables XY and $X'Y'$ are statistically close, then for most values of x, the conditional distribution $Y \mid X = x$ and $Y' \mid X' = x$ are also close.

Lemma 6.4.13. *Let X, Y, X', Y' be such that $\Delta(XY, X'Y') \leq \varepsilon$. Say that $x \in \text{supp}(X) \cap \text{supp}(X')$ is δ-*bad *if $\Delta(Y(x), Y'(x)) > \delta$. Then $\Pr[X$ is bad$] \leq 2\varepsilon/\delta$*

Proof. Suppose not, then we can derive by the triangle inequality that

$$\Delta(XY, X'Y') \geq \Delta(XY, (X, Y'(X))) - \Delta((X, Y'(X)), X'Y')$$

The second term is bounded $\Delta(X, X')$ which in turn is at most ε by hypothesis, so

$$\begin{aligned} &\geq \Delta(XY, (X, Y'(X))) - \varepsilon \\ &\geq \Pr[X \text{ bad}]\Delta(XY \mid X \text{ bad}, (X, Y'(X)) \mid X \text{ bad}) - \varepsilon \\ &> 2\varepsilon - \varepsilon \geq \varepsilon \end{aligned}$$

a contradiction. ■

The next lemma says that if two variables X, X' are close, and with high probability two dependent variables Y, Y' are also close, then the joint distributions $XY, X'Y'$ are close.

Lemma 6.4.14. *Let X, Y, X', Y' be random variables where $\Delta(X, X') \leq \varepsilon_1$. We say that $x \in \text{supp}(X) \cap \text{supp}(X')$ is ε_2-*bad *if $\Delta(Y(x), Y'(x)) \geq \varepsilon_2$, and suppose $\Pr[X$ ε_2-bad$] \leq \varepsilon_3$. Then $\Delta(XY, X'Y')) \leq \varepsilon_1 + \varepsilon_2 + \varepsilon_3$.*

Proof. This follows from the triangle inequality:

$$\begin{aligned}
\Delta(XY, X'Y') &\leq \Delta((X,Y),(X,Y'(X))) + \Delta((X,Y'(X)),(X',Y')) \\
&\leq \Delta(XY,(X,Y'(X)) + \varepsilon \\
&\leq \Pr[X \ \varepsilon_2\text{-bad}] \quad \Delta((XY \mid \ \text{bad}),\ (X,Y'(X) \mid \ \text{bad})) \\
&\quad + (1 - \Pr[X \ \varepsilon_2\text{-bad}])\varepsilon_2 + \varepsilon_1 \\
&\leq \varepsilon_3 + \varepsilon_2 + \varepsilon_1
\end{aligned}$$

■

We are now ready to prove Claim 6.4.11.

Proof of Claim 6.4.11. The base case $i = 0$ is trivial. Now assume the inductive hypothesis for $i-1$, namely that there exists $\eta_{i-1}, Y_{i-1}, Z_{i-1}$ such that

$$\Delta((\mathsf{condview}_A(\sigma_{2i-2}) \mid E_{i-1}), \mathsf{altview}_A(Y_{i-1})) \leq \delta_{i-1} \tag{6.4.3}$$

and

$$\sigma^{\mathsf{alt}}_{2i-2} = \prod_{j=1}^{i-1} \eta_j Y_{i-1} + (1 - \prod_{j=1}^{i-1} \eta_j) Z_{i-1} \tag{6.4.4}$$

Apply the i'th next message function A_i to both terms in Inequality 6.4.3 to get (because this process is identical in both cases):

$$\Delta((\mathsf{condview}_A(\sigma_{2i-1}) \mid E_{i-1}), \mathsf{altview}_A(\zeta_{2i-1})) \leq \delta_{i-1} \tag{6.4.5}$$

where for compactness we have set $\zeta_{2i-1} = Y_{i-1}A_i(\mathsf{altview}_A(Y_{i-1}))$.

Overview of remainder of proof: since $\langle A, B\rangle, \langle A', B'\rangle$ are statistically close, applying Lemma 6.4.13 and Lemma 6.4.14 means that there is little chance that, on the next message (generated by B'), the statistical distance increases by much. This allows us to apply Lemma 6.4.12, which allows us to extend the decomposition of σ_{2i} into the

part where E_i occurs and E_i does not occur into a decomposition of $\sigma_{2i}^{\mathsf{alt}}$ into parts such that the corresponding parts are also statistically close.

Applying Lemma 6.4.13: Roughly, we want to say that given $\langle A, B\rangle$, $\langle A', B'\rangle$ that are close, it is extremely unlikely that at any point that we get a partial transcript σ_{2i-1} that will cause the next message to have large statistical distance.

We know by hypothesis that $\forall i$,

$$\Delta((\mathsf{condview}_A(\sigma_{2i})), (\mathsf{condview}_A(\sigma'_{2i}))) \leq \Delta(\sigma_{2i}, \sigma'_{2i}) \leq \Delta(\langle A, B\rangle, \langle A', B'\rangle) \leq \varepsilon$$

To apply Lemma 6.4.13, set

$$X = \mathsf{condview}_A(\sigma_{2i-1}), \qquad Y = B_i(\mathsf{condview}_A(\sigma_{2i-1}))$$
$$X' = \mathsf{condview}_A(\sigma'_{2i-1}), \qquad Y' = B'_i(\sigma'_{2i-1})$$

noticing that $\mathsf{condview}_A(\sigma_{2i}) = XY$ and $\mathsf{condview}_A(\sigma'_{2i}) = X'Y'$. We say for a fixed τ_{2i-1} that $\mathsf{condview}_A(\tau_{2i-1})$ is bad if $\Delta(B_i(\mathsf{condview}_A(\tau_{2i-1})), B'_i(\tau_{2i-1})) > 2\sqrt{\varepsilon}$, and from Lemma 6.4.13 we know that, for each i, the probability that $\mathsf{condview}_A(\sigma_{2i-1})$ is bad is at most $\sqrt{\varepsilon}$. Furthermore, by hypothesis $\Pr[E_{i-1}] \geq \rho^{i-1}$ so therefore

$$\Pr[\mathsf{condview}_A(\sigma_{2i-1}) \text{ bad} \mid E_{i-1}] \leq \sqrt{\varepsilon}/\rho^{i-1} \tag{6.4.6}$$

Applying Lemma 6.4.14: Next, we want to say that because $(\sigma_{2i-1} \mid E_{i-1})$ and ζ_{2i-1} along with their views are close, and because $(\mathsf{condview}_A(\sigma_{2i-1}) \mid E_{i-1})$ is rarely bad (Inequality 6.4.6) therefore applying the next message function B'_i will not increase the distance by much. Formally, set

$$X = (\mathsf{condview}_A(\sigma_{2i-1}) \mid E_{i-1}), \qquad Y = (B_i(\mathsf{condview}_A(\sigma_{2i-1})) \mid E_{i-1})$$
$$X' = \mathsf{altview}_A(\zeta_{2i-1}), \qquad Y' = B'_i(\zeta_{2i-1})$$

Applying Lemma 6.4.14 gives us that

$$\Delta(XY, X'Y') \leq \sqrt{\varepsilon}/\rho^{i-1} + 2\sqrt{\varepsilon} + \delta_{i-1}$$

In particular, by truncating XY and $X'Y'$ to remove the random coins and only keeping the transcript, it follows that

$$\Delta((\sigma_{2i} \mid E_{i-1}), \zeta_{2i}) \leq \sqrt{\varepsilon}/\rho^{i-1} + 2\sqrt{\varepsilon} + \delta_{i-1}$$

where we have set $\zeta_{2i} = \zeta_{2i-1}B'_i(\zeta_{2i-1})$.

Applying Lemma 6.4.12: Finally, because $(\sigma_{2i} \mid E_{i-1})$ and ζ_{2i} are close, and because $(E_i \mid E_{i-1})$ happens often, we can decompose ζ_{2i} so that part of it is close to $(\sigma_{2i} \mid E_i)$. Set $X = (\sigma_{2i} \mid E_{i-1})$ and let Y, Z be the conditional distributions when E_i occurs or not. By hypothesis the conditional event $(E_i \mid E_{i-1})$ occurs with probability ρ. By Lemma 6.4.12, this means there exists $\eta_i \geq \rho - (\sqrt{\varepsilon}/\rho^{i-1} + 2\sqrt{\varepsilon} + \delta_{i-1})$ and random variables Y_i, Z_i such that $\zeta_{2i} = \eta_i Y_i + (1-\eta_i)Z_i$ such that

$$\Delta((\sigma_{2i} \mid E_i), Y_i) \leq \tfrac{3\sqrt{\varepsilon}}{2\rho^i} + \tfrac{3\sqrt{\varepsilon}}{\rho} + \tfrac{3}{2\rho}\delta_{i-1} \tag{6.4.7}$$

Setting $\delta_{i-1} = \sqrt{\varepsilon}(10/\rho)^{i-1}$ and assuming $\varepsilon = (\rho/10)^{4r}$, implies that $\eta_i \geq \rho/2$, and so the RHS of Inequality 6.4.7 is bounded by $\delta_i = \sqrt{\varepsilon}(10/\rho)^i$. Applying $\mathsf{altview}_A$ to both terms in Inequality 6.4.7 and then applying the fact that $\mathsf{condview}_A(\sigma_{2i}) = \mathsf{altview}_A(\sigma_{2i})$ conditioned on E_i (Proposition 6.4.10, Item 3) gives us our desired statement:

$$\Delta((\mathsf{condview}_A(\sigma_{2i}) \mid E_i), \mathsf{altview}_A(Y_i)) \leq \sqrt{\varepsilon}(10/\rho)^i$$

Finally, notice that using Equation 6.4.4, we get

$$\begin{aligned}
\sigma_{2i}^{\mathsf{alt}} &= \sigma_{2i-2}^{\mathsf{alt}} A_{i-1}(\sigma_{2i-2}^{\mathsf{alt}}) B'_{i-1}(\sigma_{2i-2}^{\mathsf{alt}} A_{i-1}(\sigma_{2i-2}^{\mathsf{alt}})) \\
&= \prod_{j=1}^{i-1} \eta_j Y_{i-1} A_i(Y_{i-1}) B'_i(Y_{i-1} A_i(Y_{i-1})) + (1 - \prod_{j=1}^{i-1} \eta_j) \dots \\
&= \prod_{j=1}^{i-1} \eta_j \zeta_{2i} + (1 - \prod_{j=1}^{i-1} \eta_j) \dots \\
&= \prod_{j=1}^{i} \eta_j Y_i + \prod_{j=1}^{i-1} \eta_j (1 - \eta_i) Z_i + (1 - \prod_{j=1}^{i-1} \eta_j) \dots \\
&= \prod_{j=1}^{i} \eta_j Y_i + (1 - \prod_{j=1}^{i} \eta_j) \dots
\end{aligned}$$

where "..." is a convex combination of the Y_j, Z_j's that we do not care about. Setting $\varepsilon = (\rho/10)^{4r}$ we are guaranteed that $\eta_i \geq \rho/2$ for all i, so we have that $\prod_{j=1}^{i} \eta_i \geq (\rho/2)^i = 1/\mathrm{poly}(n)$.

■

6.5 Open problems

We gave lower bounds on the key-management and cryptographic overhead of secure FL protocols. The problem of bounding the *storage* requirements in an FL protocol is also still open. Furthermore, our results here only apply to FL on *single symmetric paths* between a single sender-receiver pair. An interesting question would be to consider FL for *asymmetric paths*, where the packets Bob sends back to Alice may take a different path than the packets that Alice sends to Bob. Another interesting direction is to consider FL in networks where packets can travel simultaneously on *multiple paths*, as in the SMT framework [DDWY93].

Chapter 7

Derandomizing Chernoff bounds for matrix-valued random variables

7.1 Introduction

Chernoff bounds are extremely useful throughout theoretical computer science. Intuitively, they say that a random sample approximates the average, with a probability of deviation that goes down exponentially with the number of samples. Typically we are concerned with real-valued random variables, but recently several applications have called for large-deviation bounds for matrix-valued random variables. Such a bound was given by Ahlswede and Winter [AW02] (see Theorem 7.2.6 and Theorem 7.2.8 for a precise statement of their bounds).

In particular, the matrix-valued bound seems useful in giving new proofs of probabilistic constructions of expander graphs [AR94] and also in the randomized rounding of semi-definite covering problems, with further applications in quantum information theory [AW02].

In this chapter we use the method of pessimistic estimators, originally formulated in [Rag88], to derandomize the Chernoff bound of [AW02], and in the process derandomize the Alon-Roichman theorem and the randomized rounding of covering SDP's. Pessimistic estimators succeeded earlier work on the method of conditional probabilities, which was already described in the first edition of [Spe94]. Ideas similar to those of [Rag88] also appeared in [BS84].

Arora and Kale [AK07] independently reached results similar to the ones presented in this chapter that imply the applications to constructing expanding Cayley graphs and semi-definite covering programs.

The chapter is organized as follows. In Section 7.2 we define the linear algebra notation we use and prove the Chernoff bounds of Ahlswede-Winter, given in Theorem 7.2.6 and Theorem 7.2.8. In Section 7.3 we review the method of pessimistic estimators and how it is used to derandomize algorithms. In Section 7.4 we construct pessimistic estimators for the Ahlswede-Winter Chernoff bounds. We apply these estimators to derandomize the construction of Cayley expanders in Section 7.5 and to derandomize the rounding of integer covering SDP's in Section 7.6. We then state a generalization of our main theorem to the abstract setting of finite-dimensional Hilbert spaces.

7.1.1 Historical notes

The results in this chapter were originally published in [WX08].

7.2 Matrix-valued random variables and Ahlswede-Winter's Chernoff Bound

We will work with real symmetric $d \times d$ matrices, which we will denote $\mathcal{M}_d$. We let I_d denote the identity matrix in $\mathcal{M}_d$, and will write simply I when the dimension is clear. For any $A \in \mathcal{M}_d$ we let $\lambda_1(A) \geq \ldots \geq \lambda_d(A)$ denote the eigenvalues of A in non-increasing order. Recall that every matrix $A \in \mathcal{M}_d$ is diagonalizable in an orthonormal basis.

We will measure distance between matrices using the matrix norm $\|A\| = \max_v \|Av\|/\|v\| = \max_i |\lambda_i(A)|$. We will also frequently use the trace, $\mathrm{Tr}(A) = \sum_{i=1}^d \lambda_i(A)$. It is well-known that for any orthonormal basis $v_1, \ldots, v_d \in \mathbb{R}^d$ we have that $\mathrm{Tr}(A) = \sum_{i=1}^d \langle v_i, Av_i \rangle$, where $\langle \cdot, \cdot \rangle$ denotes the usual inner product over $\mathbb{R}^d$.

We say that a matrix $A \in \mathcal{M}_d$ is positive semi-definite (p.s.d.) if all its eigenvalues are non-negative. We will use the fact that A is p.s.d. iff for all $v \in \mathbb{R}^d$, $\langle v, Av \rangle \geq 0$. We let $A \geq 0$ denote that A is p.s.d. We use the ordering of symmetric matrices given by this definition, namely $A \leq B$ iff $B - A \geq 0$. For two matrices $A \leq B$, we will let $[A, B]$ denote the set of all symmetric matrices C such that $A \leq C$ and $C \leq B$.

We will work with the matrix exponential, which is defined by

$$\exp(A) = \sum_{\ell=0}^{\infty} \frac{A^\ell}{\ell!}$$

Recall that the matrix exponential is convergent for all matrices. Furthermore, it is not hard to see for $A \in \mathcal{M}_d$ that $\exp(A)$ is diagonalizable in the same basis as A, and that $\lambda_i(\exp(A)) = e^{\lambda_i(A)}$ for all $1 \leq i \leq d$. Also, for all $A \in \mathcal{M}_d$, it holds that $\exp(A) \geq 0$.

We will consider matrix-valued random variables of the following form. We let $f : [n] \to [-I_d, I_d]$, where $[n] = \{1, \ldots, n\}$. Let X be a distribution (not neces-

sarily uniform) over $[n]$, and consider the variable $f(X)$. This is a natural extension of bounded discrete random variables over the reals, which may be thought of as functions $f : [n] \to [-1, 1]$. We will let the expectation of $f(X)$ be the obvious thing: $\mathbb{E}[f(X)] = \sum_{i=1}^{n} \Pr[X = i] f(i)$. Note that because Tr is linear, $\mathbb{E}$ and Tr commute: $\mathbb{E}[\mathrm{Tr}(f(X))] = \mathrm{Tr}(\mathbb{E}[f(X)])$. We let $\mathrm{supp}(X)$ denote the set of all values of X that occur with non-zero probability. When we say that something holds for a random variable X *always*, we mean that it holds for every element in $\mathrm{supp}(X)$.

We will use the following useful facts several times:

Fact 7.2.1. *If $A, B \in \mathcal{M}_d$ and $B \geq 0$, then $\mathrm{Tr}(AB) \leq \|A\|\mathrm{Tr}(B)$.*

Proof. Let $v_1, \ldots, v_d$ be the orthonormal diagonal basis of A, with corresponding eigenvalues $\lambda_i = \lambda_i(A)$. Then we may write

$$\begin{aligned}\mathrm{Tr}(AB) &= \sum_{i=1}^{d} \langle v_i, ABv_i \rangle \\ &= \sum_{i=1}^{d} \lambda_i \langle v_i, Bv_i \rangle\end{aligned}$$

Since $B \geq 0$ we know that $\langle v_i, Bv_i \rangle \geq 0$, so we get

$$\begin{aligned}&\leq \sum_{i=1}^{d} \max_j \lambda_j \langle v_i, Bv_i \rangle \\ &\leq \|A\|\mathrm{Tr}(B)\end{aligned}$$

■

Theorem 7.2.2 (Golden-Thompson inequality, [Gol65, Tho65])**.** *For $A, B \in \mathcal{M}_d$, we have*

$$\mathrm{Tr}(\exp(A + B)) \leq \mathrm{Tr}(\exp(A)\exp(B))$$

The proof of this is outside the scope of this chapter.

Ahlswede and Winter introduce a generalization of Markov's inequality for matrix-valued random variables.

Theorem 7.2.3 (Markov's inequality [AW02]). *For any $\gamma > 0$, any function $g : [n] \to \mathcal{M}_d$ such that $g(x) \geq 0$ for all $x \in [n]$, and for any random variable X over $[n]$, we have*

$$\Pr[g(X) \not\leq \gamma I] \leq \tfrac{1}{\gamma}\mathrm{Tr}(\mathbb{E}[g(X)])$$

Proof.

$$\begin{aligned}\Pr[g(X) \not\leq \gamma I] &= \Pr[\|g(X)\| > \gamma] \\ &\leq \tfrac{1}{\gamma}\mathbb{E}[\|g(X)\|]\end{aligned}$$

Since $g(X) \geq 0$ always, we have $\|g(X)\| \leq \mathrm{Tr}(g(X))$ always, so we get:

$$\begin{aligned}&\leq \tfrac{1}{\gamma}\mathbb{E}[\mathrm{Tr}(g(X))] \\ &= \tfrac{1}{\gamma}\mathrm{Tr}(\mathbb{E}[g(X)])\end{aligned}$$

■

The following Theorem 7.2.4 is the main theorem proving [AW02]'s Chernoff-type bound. We will use Theorem 7.2.4, which holds for all distributions, to derive two corollaries (Theorem 7.2.6 and Theorem 7.2.8), which hold for more specific kinds of distributions. In addition, the proof of Theorem 7.2.4 will give us the pessimistic estimators corresponding to the two corollaries.

Theorem 7.2.4 ([AW02]). *Suppose $f : [n] \to [-I_d, I_d]$ and let $X_1, \ldots, X_k$ be arbitrary independent random variables distributed over $[n]$. Then for all $\gamma \in \mathbb{R}$:*

$$\Pr[\tfrac{1}{k}\sum_{j=1}^{k} f(X_j) \not\leq \gamma I] \leq d e^{-t\gamma k} \prod_{j=1}^{k} \|\mathbb{E}[\exp(t f(X_j))]\|$$

Proof. The proof begins analogously to the real-valued case, generalizing the classical Bernstein trick. We first multiply by an optimization constant $t > 0$ and exponentiate to obtain

$$\Pr[\tfrac{1}{k}\sum_{j=1}^{k} f(X_j) \not\preceq \gamma I] = \Pr[\exp(t\sum_{j=1}^{k} f(X_j)) \not\preceq e^{t\gamma k} I]$$

The equality holds because for any $A \in \mathcal{M}_d, \alpha \in \mathbb{R}$, the statement $A \not\preceq \alpha I$ is equivalent to saying some eigenvalue of A is larger than α, which is the same as saying that some eigenvalue of $\exp(A)$ is larger than e^{α}, which in turn is equivalent to $\exp(A) \not\preceq e^{\alpha} I$. Then the following inequality is a direct consequence of Theorem 7.2.3 since $\exp(A) \geq 0$ for all $A \in \mathcal{M}_d$.

$$\Pr[\tfrac{1}{k}\sum_{j=1}^{k} f(X_j) \not\preceq \gamma I] \leq e^{-t\gamma k}\mathrm{Tr}(\mathbb{E}[\exp(t\sum_{j=1}^{k} f(X_j))]) \tag{7.2.1}$$

Then we apply Fact 7.2.1 and the Golden-Thompson Inequality Theorem 7.2.2 to bound the expression in a manageable form. This step will be expressed in the following lemma.

Lemma 7.2.5. *For any matrix $A \in \mathcal{M}_d$, any $f : [n] \to \mathcal{M}_d$ and any random variable X over $[n]$, we have*

$$\mathrm{Tr}(\mathbb{E}_X[\exp(A + f(X))]) \leq \|\mathbb{E}[\exp(f(X))]\| \cdot \mathrm{Tr}(\exp(A))$$

To obtain Theorem 7.2.4, we simply apply Lemma 7.2.5 to Inequality 7.2.1 repeatedly:

$$\begin{aligned}
&\Pr[\tfrac{1}{k}\textstyle\sum_{j=1}^{k} f(X_j) \not\preceq \gamma I] \leq e^{-t\gamma k}\mathrm{Tr}(\mathbb{E}[\exp(t\textstyle\sum_{j=1}^{k} f(X_j))]) \\
&= e^{-t\gamma k}\mathbb{E}_{X_1,\ldots,X_{k-1}}\left[\mathrm{Tr}(\mathbb{E}_{X_k}[\exp(t\textstyle\sum_{j=1}^{k-1} f(X_j) + tf(X_k))])\right] && \textit{(Independence)} \\
&\leq e^{-t\gamma k}\mathbb{E}_{X_1,\ldots,X_{k-1}}\left[\|\mathbb{E}[\exp(tf(X_k))]\| \cdot \mathrm{Tr}(\exp(t\textstyle\sum_{j=1}^{k-1} f(X_j)))\right] && \textit{(Lemma 7.2.5)} \\
&= e^{-t\gamma k}\|\mathbb{E}[\exp(tf(X_k))]\| \cdot \mathrm{Tr}(\mathbb{E}_{X_1,\ldots,X_{k-1}}[\exp(t\textstyle\sum_{j=1}^{k-1} f(X_j))]) && (\mathbb{E},\ \mathrm{Tr}\ \textit{commute}) \\
&\leq e^{-t\gamma k}\textstyle\prod_{j=1}^{k}\|\mathbb{E}[\exp(tf(X_j))]\|\mathrm{Tr}(I) && \textit{(Repeat } k \textit{ times)} \\
&= de^{-t\gamma k}\textstyle\prod_{j=1}^{k}\|\mathbb{E}[\exp(tf(X_j))]\|
\end{aligned} \tag{7.2.2}$$

This completes the proof modulo Lemma 7.2.5. ■

Proof of Lemma 7.2.5.

$$\begin{aligned}
\mathrm{Tr}(\mathbb{E}[\exp(A + f(X))]) &= \mathbb{E}[\mathrm{Tr}(\exp(A + f(X)))] && (\mathbb{E},\ \mathrm{Tr}\ \textit{commute}) \\
&\leq \mathbb{E}[\mathrm{Tr}(\exp(f(X))\exp(A))] && \textit{(Golden-Thompson)} \\
&\leq \mathrm{Tr}(\mathbb{E}[\exp(f(X))]\exp(A)) && (\mathbb{E},\ \mathrm{Tr}\ \textit{commute}) \\
&\leq \|\mathbb{E}[\exp(tf(X))]\| \cdot \mathrm{Tr}(\exp(A)) && \textit{(Fact 7.2.1)}
\end{aligned}$$

■

Now we will draw two corollaries from this main theorem. These two corollaries are useful in different settings; the first guarantees that the probability of an additive deviation is small, while the second that of a multiplicative deviation.

Theorem 7.2.6 ([AW02])**.** *Let $f : [n] \to [-I_d, I_d]$. Let X be distributed over $[n]$ with $\mathbb{E}_X[f(X)] = 0$, and let $X_1, \ldots, X_k$ be i.i.d. copies of X. Then for all $1 > \gamma > 0$:*[1]

$$\Pr[\tfrac{1}{k}\sum_{i=1}^{k} f(X_i) \not\preceq \gamma I] \leq de^{-\gamma^2 k/4}$$

[1]For the sake of simplicity, no attempt was made to optimize the constant in the exponent of the bound in this analysis. To get a tighter bound, we can apply the analysis of [AW02] to get a bound

Note that the other direction $\frac{1}{k}\sum_{i=1}^{k} f(X_i) \not\geq -\gamma I$ holds with the same bound by considering $-f$.

Proof. We require only Theorem 7.2.4 and a simple claim. Because all the X_i are i.i.d. Theorem 7.2.4 gives us

$$\Pr[\tfrac{1}{k}\sum_{i=1}^{k} f(X_i) \not\leq \gamma I] \leq d e^{-t\gamma k} \|\mathbb{E}[\exp(tf(X))]\|^k$$

We use the following claim to bound the RHS.

Claim 7.2.7. *$\|\mathbb{E}[\exp(tf(X))]\| \leq 1 + t^2$ for $t \leq 1/2$.*

Proof. This follows from the Taylor expansion of exp:

$$\begin{aligned}\|\mathbb{E}[\exp(tf(X))\| &= \|\mathbb{E}[I + tf(X) + \tfrac{(tf(X))^2}{2} + \ldots]\| \\ &= \|I + t\mathbb{E}[f(X)] + \mathbb{E}[(tf(X))^2/2 + \ldots]\|\end{aligned}$$

Since $\mathbb{E}[f(X)] = 0$, applying the triangle inequality, and using $\|f(X)\| \leq 1$ always, we have

$$\leq 1 + \sum_{\ell=2}^{\infty} t^\ell/\ell!$$

Since $t = \gamma/2 \leq 1/2$ this gives

$$\leq 1 + t^2$$

■

of $de^{-kD(\frac{1+\gamma}{2}\|\frac{1}{2})}$. Here $D(p\|q) = p(\log p - \log q) + (1-p)(\log(1-p) - \log(1-q))$ is the relative entropy function, and using the approximation $D(\frac{1+\gamma}{2}\|\frac{1}{2}) \geq \gamma^2/(2\ln 2)$, which can be shown by looking at the Taylor expansion of $D(\cdot\|\cdot)$, we have the improved bound of $de^{-k\gamma^2/(2\ln 2)}$

We will choose $t = \gamma/2 \leq 1/2$, so we may apply Claim 7.2.7 to Theorem 7.2.4 to get

$$\begin{aligned}
\Pr[\tfrac{1}{k}\textstyle\sum_{i=1}^{k} f(X_i) \not\leq \gamma I] &\leq d e^{-t\gamma k}(1+t^2)^k \\
&\leq d e^{-t\gamma k + t^2 k} && \textit{(Using } 1+x \leq e^x \textit{ for all } x \in \mathbb{R}\textit{)} \\
&\leq d e^{-\gamma^2 k/4} && \textit{(Choosing } t = \gamma/2\textit{)}
\end{aligned}$$

■

Theorem 7.2.8 ([AW02])**.** *Let $f : [n] \to [0, I_d]$. Let X be distributed over $[n]$, with $M = \mathbb{E}_X[f(X)] \geq \mu I$ for some $\mu \in (0,1)$. Let $X_1, \ldots, X_k$ be i.i.d. copies of X. Then we have, for all $\gamma \in [0, 1/2]$,*

$$\Pr[\tfrac{1}{k}\sum_{i=1}^{k} f(X_i) \not\geq (1-\gamma)\mu I] \leq d e^{-\gamma^2 \mu k/(2\ln 2)}$$

Proof. We can assume without loss of generality that $M = \mu I$, for if not, we could work with $g(x) = \mu M^{-1/2} f(x) M^{-1/2}$ instead. Because the direction of this bound is the opposite of what we proved in Theorem 7.2.4, we will work with $I - f$ to get:

$$\Pr[\tfrac{1}{k}\sum_{i=1}^{k} f(X_i) \not\geq (1-\gamma)\mu I] = \Pr[\tfrac{1}{k}\sum_{i=1}^{k} (I - f(X_i)) \not\leq (1-(1-\gamma)\mu) I] \tag{7.2.3}$$

Applying Theorem 7.2.4

$$\leq d e^{-t(1-(1-\gamma)\mu)k} \|\mathbb{E}[\exp(t(I - f(X)))]\|^k \tag{7.2.4}$$

$$= d \|\mathbb{E}[\exp(-tf(X)) e^{t(1-\gamma)\mu}]\|^k \tag{7.2.5}$$

This last quantity was analyzed in the proof of Theorem 19 of [AW02], with the following conclusion which we state without proof:

Claim 7.2.9 ([AW02])**.** *For $t = \log(\frac{1-(1-\gamma)\mu}{1-\mu}\frac{1}{(1-\gamma)})$, we have*

$$\|\mathbb{E}[\exp(-tf(X))] e^{t(1-\gamma)\mu}\| \leq e^{-\gamma^2 \mu/(2\ln 2)}$$

Applying this claim to Inequality 7.2.5 gives us the theorem. ■

7.3 Method of pessimistic estimators

First we review the method of pessimistic estimators, due to Raghavan [Rag88]. The setting is the following: we have a random variable X and we know that with some non-zero probability an event $\sigma(X)$ occurs, i.e. $\Pr[\sigma(X) = 1] > 0$, where $\sigma : \text{supp}(X) \to \{0, 1\}$, $\sigma(x) = 1$ iff x is in the event. We wish to efficiently and deterministically find a particular $x \in \text{supp}(X)$ such that $\sigma(x) = 1$.

Our application of pessimistic estimators is to derandomizing probabilistic algorithms. In particular, suppose we have a randomized algorithm that constructs an object, and with some non-zero probability that object satisfies some property. Thus, our event σ is the event that the object satisfies the property, and our goal is to deterministically and efficiently find the object. In this chapter our two main applications are to deterministically and efficiently find a small generating set of a group that satisfies expansion, and to find an integer solution to a SDP covering problem that satisfies feasibility and some approximation guarantee. Both problems were previously known to have randomized algorithms, and we use our pessimistic estimators to derandomize these algorithms.

We will only be concerned with random variables with finite state space with a product structure, and we will sub-divide the variable into many parts. Thus we use the notation $\vec{X}$ to denote a random variable where *w.l.o.g.* $\text{supp}(\vec{X}) \subseteq [n]^k$ for some $k, n \in \mathbb{N}$ (these will be chosen according to the application). Let $\vec{X} = (X_1, \ldots, X_k)$, where each $X_i \in [n]$. To find a "good" setting of $\vec{X}$, we will iteratively find settings of X_1, then X_2, and so forth until we have a complete setting of $\vec{X}$.

By the definition of expectation

$$\Pr_{\vec{X}}[\sigma(\vec{X}) = 0] = \mathbb{E}_{X_1}[\Pr[\sigma(\vec{X}) = 0 \mid X_1]]$$

Now by averaging there must exist at least one setting $x_1 \in [n]$ of X_1 such that

$$\Pr[\sigma(\vec{X}) = 0 \mid X_1 = x_1] \leq \mathbb{E}_{X_1}[\Pr[\sigma(\vec{X}) = 0 \mid X_1]]$$

We set $X_1 = x_1$, and then repeat the same reasoning for $X_2, \ldots X_k$. Let us denote the resulting setting of $\vec{X}$ by $\vec{x}$. Thus at the end we have $\Pr[\sigma(\vec{x}) = 0] \leq \Pr[\sigma(\vec{X}) = 0]$. But note that we supposed that $\Pr[\sigma(\vec{X}) = 0] < 1$, and since $\vec{x}$ is a *fixed* vector, it must be that $\Pr[\sigma(\vec{x}) = 0] = 0$ and therefore $\sigma(\vec{x}) = 1$.

The difficulty with turning this into an algorithm is in calculating the probabilities, for each $1 \leq i \leq k$ and, $\forall x_1, \ldots, x_i \in [n]$

$$\Pr_{X_{i+1},\ldots,X_k}[\sigma(\vec{X}) = 0 \mid X_1 = x_1, \ldots, X_i = x_i]$$

since they may not be efficiently computable. The following definition circumvents this problem.

Definition 7.3.1. Let $\sigma : [n]^k \to \{0, 1\}$ be an event on a random variable $\vec{X}$ distributed over $[n]^k$ and suppose $\Pr[\sigma(\vec{X}) = 1] > 0$. We say that $\phi_0, \ldots, \phi_k$, $\phi_i : [n]^i \to [0, 1]$ (here ϕ_0 is just a number in $[0, 1]$), are *pessimistic estimators* for σ if the following hold.

1. For any i and any fixed $x_1, \ldots, x_i \in [n]$, we have that

$$\Pr_{X_{i+1},\ldots,X_k}[\sigma(x_1, \ldots, x_i, X_{i+1}, \ldots, X_k) = 0] \leq \phi_i(x_1, \ldots, x_i)$$

2. For any i and any fixed $x_1, \ldots, x_i \in [n]$:

$$\mathbb{E}_{X_{i+1}}\phi_{i+1}(x_1, \ldots, x_i, X_{i+1}) \leq \phi_i(x_1, \ldots, x_i)$$

Our definition is stronger than the standard definition of pessimistic estimators, in that in the second condition usually all that is required is for all $x_1, \ldots, x_i \in [n]$, there exists $x_{i+1} \in [n]$ such that $\phi_{i+1}(x_1, \ldots, x_{i+1}) \leq \phi_i(x_1, \ldots, x_i)$. Our estimators satisfy this stronger definition and we will find it useful, especially when composing estimators (see Lemma 7.3.3).

We will also want the pessimistic estimators to be *efficient*, namely each ϕ_i is efficiently computable, and *useful*, which means $\phi_0 < 1$. This last condition is because ϕ_0 is a bound on the initial probability of failure, which we need to be strictly less than 1.

Theorem 7.3.2 ([Rag88])**.** *If there exist efficient and useful pessimistic estimators* $(\phi_0, \ldots, \phi_k)$ *for an event* σ*, then one can efficiently compute a* fixed $\vec{x} \in [n]^k$ *such that* $\sigma(\vec{x}) = 1$.

Proof. We pick $x_1, \ldots, x_k$ one by one. At step 0 we have $\phi_0 < 1$ since the estimators are useful.

At step i, we have $x_1, \ldots, x_i$ already fixed. Enumerate over $x_{i+1} \in [n]$ and choose the value such that $\phi_{i+1}(x_1, \ldots, x_{i+1}) \leq \phi_i(x_1, \ldots, x_i) < 1$. We are guaranteed that

$$\mathbb{E}_{X_{i+1}}[\phi_{i+1}(x_1, \ldots x_i, X_{i+1})] \leq \phi_i(x_1, \ldots, x_i)$$

by property 2 of Definition 7.3.1, and so by averaging there must exist a fixed $x_{i+1} \in [n]$ that is at most the expectation on the LHS of the above inequality. We can compute the value of the estimator efficiently by hypothesis.

Finally, we have after k steps that $\phi_k(\vec{x}) < 1$ and by property 1 we have that $\Pr[\sigma(\vec{x}) = 0] \leq \phi_k(\vec{x}) < 1$, and therefore $\sigma(\vec{x}) = 1$.

The algorithm runs through k steps, and each step is efficient, so the overall algorithm is efficient. ∎

We will find it useful to compose estimators, which is possible from the following lemma.

Lemma 7.3.3. *Suppose $\sigma, \tau : [n]^k \to \{0,1\}$ are events on $\vec{X}$, which is distributed over $[n]^k$. Suppose that $(\phi_0, \ldots, \phi_k), (\psi_0, \ldots, \psi_k)$ are pessimistic estimators for σ, τ respectively. Then $(\phi_0 + \psi_0, \ldots, \phi_k + \psi_k)$ are pessimistic estimators for the event $\sigma \cap \tau$.*

Proof. We need to verify the properties of Definition 7.3.1.

1. This is verified by a union bound:

$$\begin{aligned}
&\Pr[(\sigma \cap \tau)(x_1, \ldots, x_i, X_{i+1}, \ldots, X_k) = 0] \\
&\quad \leq \Pr[\sigma(x_1, \ldots, x_i, X_{i+1}, \ldots, X_k) = 0] \\
&\qquad + \Pr[\tau(x_1, \ldots, x_i, X_{i+1}, \ldots, X_k) = 0] \\
&\quad \leq (\phi_i + \psi_i)(x_1, \ldots, x_i)
\end{aligned}$$

2. This is immediate from linearity of expectation.

■

7.4 Applying pessimistic estimators

The method of pessimistic estimators extends to the AW Chernoff bound. We will first describe pessimistic estimators for Theorem 7.2.6 and then for Theorem 7.2.8. They are essentially identical except for the difference in distributions in the two settings, and the proofs that the pessimistic estimators satisfy Definition 7.3.1 rely mainly on Lemma 7.2.5. In both cases, they allow us to efficiently and deterministically find settings $x_1, \ldots, x_k$ such that bad event bounded by Theorem 7.2.6 (resp. Theorem 7.2.8) does not occur.

Theorem 7.4.1. *Let $f : [n] \to [-I_d, I_d]$. Let X be distributed over $[n]$ with $\mathbb{E}_X[f(X)] = 0$, and let $X_1, \ldots, X_k$ be i.i.d. copies of X. Fix $1 > \gamma > 0$. Let $t = \gamma/2$. Suppose that $\mathbb{E}[\exp(tf(X))]$ is efficiently computable.*

Combining the notation of Section 7.2 and Section 7.3, we let $\vec{X} = (X_1, \ldots, X_k)$ with $X_i \in [n]$ and we let $\sigma : [n]^k \to \{0, 1\}$ be the event $\sigma(\vec{x}) = 1$ if $\frac{1}{k}\sum_{i=1}^{k} f(x_i) \leq \gamma I$ and $\sigma(\vec{x}) = 0$ otherwise. Then the following $(\phi_0, \ldots, \phi_k), \phi_i : [n]^i \to [0, 1]$ are efficient pessimistic estimators *for σ.*

$$\phi_0 = d e^{-t\gamma k} \|\mathbb{E}[\exp(tf(X))]\|^k \quad (\textit{which is at most } d e^{-\gamma^2 k/4})$$

$$\phi_i(x_1, \ldots, x_i) = d e^{-t\gamma k} \mathrm{Tr}(\exp(t \sum_{j=1}^{i} f(x_j))) \cdot \|\mathbb{E}[\exp(tf(X))]\|^{k-i}$$

Proof. We verify the properties of Definition 7.3.1.

1. From Inequality 7.2.1:

$$\begin{aligned}\Pr[\tfrac{1}{k}\sum_{i=1}^{k} f(X_i) \not\leq \gamma I] &\leq d e^{-t\gamma k} \mathrm{Tr}(\mathbb{E}[\exp(t \sum_{j=1}^{k} f(X_j))]) \\ &\leq d e^{-t\gamma k} \mathrm{Tr}(\mathbb{E}[\exp(t \sum_{j=1}^{i} f(X_j))]) \prod_{j=i+1}^{k} \|\mathbb{E}[\exp(tf(X_j))]\|\end{aligned}$$

   By fixing $X_j = x_j$ for all $j \leq i$, we derive that

$$\begin{aligned}&\Pr[\tfrac{1}{k}\sum_{i=1}^{k} f(X_i) \not\leq \gamma I \mid X_1 = x_1, \ldots, X_i = x_i] \\ &\leq d e^{-t\gamma k} \mathrm{Tr}(\exp(t \sum_{j=1}^{i} f(x_j))) \cdot \|\mathbb{E}[\exp(tf(X))]\|^{k-i} \\ &= \phi_i(x_1, \ldots, x_i)\end{aligned}$$

2. We use the following derivation, where the inequality follows from Lemma 7.2.5:

$$\begin{aligned}
&\mathbb{E}_{X_{i+1}}[\phi_{i+1}(x_1,\ldots,x_i,X_{i+1})] \\
&\quad = de^{-t\gamma k}\mathrm{Tr}(\mathbb{E}_{X_{i+1}}[\exp(t\sum_{j=1}^{i} f(x_i) + tf(X_{i+1}))]) \cdot \|\mathbb{E}(\exp(tf(X)))\|^{k-i-1} \\
&\quad \leq de^{-t\gamma k}\mathrm{Tr}(\exp(t\sum_{j=1}^{i} f(x_i))) \cdot \|\mathbb{E}(\exp(tf(X)))\|^{k-i} \\
&\quad = \phi_i(x_1,\ldots,x_i)
\end{aligned}$$

To see that the ϕ_i are efficiently computable, we will specify the input to the algorithm as a function f (which we assume is given as a list of $d \times d$ matrices $f(1),\ldots,f(n)$) and 1^k. Thus we desire the algorithm to be computable in time $\mathrm{poly}(n,d,k)$. We require multiplication, addition, trace, matrix exponential, and norm computations. The first three are obviously efficient; the last two are efficient because eigenvalues of a $d \times d$ matrix can be computed (and hence it can be diagonalized thus making the exponential and norm computations trivial) in $O(d^3)$ numerical operations [GL89]. On a machine with finite precision, we can truncate the estimators to a sufficiently fine resolution so that the truncated estimators behave essentially as the real-valued estimators do.

■

Theorem 7.4.1 gives us pessimistic estimators $(\phi_0,\ldots,\phi_k)$ for σ, and the same proof gives efficient pessimistic estimators $(\psi_0,\ldots,\psi_k)$ for the event $\tau(\vec{x}) = 1$ iff $\frac{1}{k}\sum_{i=1}^{k} f(x_i) \geq -\gamma I$ by applying Theorem 7.2.6 to $-f$. Combining these with the ϕ_i gives us the following.

Corollary 7.4.2. *Let $f : [n] \to [-I_d, I_d]$. Let X be distributed over $[n]$ with $\mathbb{E}_X[f(X)] = 0$, and let $X_1,\ldots,X_k$ be i.i.d. copies of X. Fix $1 > \gamma > 0$ and fix $t = \gamma/2$. Suppose that $\mathbb{E}[\exp(tf(X))]$ and $\mathbb{E}[\exp(-tf(X))]$ are efficiently computable.*

Let $\eta : [n]^k \to \{0,1\}$ *be the event* $\eta(\vec{x}) = 1$ *if* $\|\frac{1}{k}\sum_{i=1}^{k} f(x_i)\| \leq \gamma$ *and* $\eta(\vec{x}) = 0$ *otherwise. Then* $(\phi_0 + \psi_0, \ldots, \phi_k + \psi_k)$ *are efficient pessimistic estimators for* η*.*

Proof. Note that $\eta = \sigma \cap \tau$. Efficiency is clear. We can apply Lemma 7.3.3 to get that $(\phi_0 + \psi_0, \ldots, \phi_k + \psi_k)$ is a pessimistic estimator for the event $\eta = \sigma \cap \tau$. ■

This allows us to derandomize Theorem 7.2.6 efficiently. Notice that in general the only property of X that we need is to be able to compute $\mathbb{E}[\exp(tf(X))]$ and $\mathbb{E}[\exp(-tf(X))]$.This is of course true when X is uniform, or when we can efficiently compute $\Pr[X = x]$ for each $x \in [n]$. The actual distribution is irrelevant, since we exhaustively search through the entire space for the choice of each X_i.

Theorem 7.4.3. *Let* $f : [n] \to [-I_d, I_d]$ *be such that there exists a distribution* X *over* $[n]$ *such that* $\mathbb{E}[f(X)] = 0$*. Then for* $k = O(\frac{1}{\gamma^2}\log d)$*, we can efficiently and deterministically find* $\vec{x} \in [n]^k$ *such that* $\|\frac{1}{k}\sum_{i=1}^{k} f(x_i)\| \leq \gamma$*.*

Proof. Use the efficient pessimistic estimators of Corollary 7.4.2. Pick $k = O(\frac{1}{\gamma^2}\log d)$ such that $\phi_0 + \psi_0 < 1$ and so that the estimators are useful. We may then apply Theorem 7.3.2 to get the result. ■

We can construct pessimistic estimators for Theorem 7.2.8 in the same way.

Theorem 7.4.4. *Let* $f : [n] \to [0, I_d]$*. Let* X *be distributed over* $[n]$*, with* $M = \mathbb{E}_X[f(X)] \geq \mu I$ *for some* $\mu \in (0,1)$*. Let* $X_1, \ldots, X_k$ *be i.i.d. copies of* X*. Fix* $t = \log(\frac{1-(1-\gamma)\mu}{1-\mu}\frac{1}{(1-\gamma)})$*.*

Let $\vec{X} = (X_1, \ldots, X_k)$ *with* $X_i \in [n]$ *and we let* $\sigma : [n]^k \to \{0,1\}$ *be the event* $\sigma(\vec{x}) = 1$ *if* $\frac{1}{k}\sum_{i=1}^{k} f(x_i) \geq (1-\gamma)\mu I$ *and* $\sigma(\vec{x}) = 0$ *otherwise. Then the following*

$(\phi_0, \ldots, \phi_k), \phi_i : [n]^i \to [0,1]$ *are efficient pessimistic estimators for* σ*.*

$$\phi_0 = d e^{tk(1-\gamma)\mu} \|\mathbb{E}[\exp(-tf(X))]\|^k \quad (\textit{which is at most } d e^{-\gamma^2 \mu k/(2\ ln 2)})$$

$$\phi_i(x_1, \ldots, x_i) = d e^{tk(1-\gamma)\mu} \mathrm{Tr}(\exp(-t \sum_{j=1}^{i} f(x_j))) \cdot \|\mathbb{E}[\exp(-tf(X))]\|^{k-i}$$

Proof. The proof follows exactly along the lines of Theorem 7.4.1. ■

Theorem 7.4.5. *Let* $f : [n] \to [0, I_d]$ *be such that there exists a distribution* X *over* $[n]$ *and a number* $\mu \in (0,1)$ *such that* $\mathbb{E}[f(X)] \geq \mu I$*. Then for* $k = O(\frac{1}{\gamma^2 \mu} \log d)$*, we can efficiently and deterministically find* $\vec{x} \in [n]^k$ *such that* $\frac{1}{k} \sum_{i=1}^{k} f(x_i) \geq (1-\gamma)\mu I$*.*

Proof. Use the efficient pessimistic estimators of Theorem 7.4.4, and notice for our choice of k that $\phi_0 < 1$ so they are useful. Then apply Theorem 7.3.2. ■

7.5 $O(\log n)$ expanding generators for any group

Our main application is a complete derandomization of the Alon-Roichman [AR94] theorem, which states that a certain kind of expander graph may be constructed by random sampling (details below). Expander graphs have a central role in theoretical computer science, especially in but not limited to the study of derandomization. Indeed, they have found a large number of applications in a variety of areas such as deterministic amplification [CW89, IZ89], security amplification in cryptography [GIL+90], hardness of approximation [ALM+98, AFWZ95], extractor construction (e.g. see surveys [NT99, Gol97, Sha02]), construction of efficient error-correcting codes [Spi95, BH04], construction of ε-biased spaces [NN93] and much more. See [HLW06] for a comprehensive survey.

We derandomize the proof of the Alon-Roichman theorem given by [LR04] (see also [LS04]) to give a deterministic and efficient construction of the expanding generat-

ing set. We show how it implies an optimal solution to a problem of Shpilka and Wigderson [SW04] (see also [GS02]), significantly improving their results.

7.5.1 Definitions

Given a connected undirected d-regular graph $G = (V, E)$ on n vertices, we define its normalized adjacency matrix A, $A_{ij} = e_{ij}/d$ where e_{ij} is the number of edges between vertices i and j (we allow self-loops and multiple edges). It is easy to see that A is real and symmetric.

It is well-known that the set of eigenvalues of A is of the form $1 = \lambda_1(A) > \lambda_2(A) \geq \ldots \geq \lambda_n(A)$. Note the strict separation between $\lambda_1(A)$ and $\lambda_2(A)$, which follows from connectivity. The eigenvalues of G are the eigenvalues of A. Note that 1 is an eigenvalue of multiplicity 1, and with corresponding eigenvector $u = [1/\sqrt{n}, \ldots, 1/\sqrt{n}]^T$, which we call the uniform vector. Alternatively, the eigenvalue 1 also corresponds to the uniform eigenspace, given by the matrix J/n, where J is the all 1's matrix, which is the orthogonal projection onto the space spanned by the eigenvector u.

The Cayley graph $\mathrm{Cay}(H; S)$ on a group H with respect to the generating multi-set $S \subset H$ is the graph whose vertex set is H, and where h and h' are connected by an edge if there exists $s \in S$ such that $h' = hs$ (allowing for multiple edges for multiple elements in S). We require S to be symmetric, namely for each $s \in S$, we also have $s^{-1} \in S$ (this is to make the graph undirected). Let $\lambda(\mathrm{Cay}(H; S))$ denote the second-largest eigenvalue (in absolute value) of the normalized adjacency matrix of the Cayley graph.

Our goal is to construct an algorithm that, for a fixed $\gamma < 1$, takes as input the multiplication table of a group H of size n and efficiently constructs a small generating set S such that $\lambda(\mathrm{Cay}(H; S)) < \gamma$. This is given by the following theorem.

Theorem 7.5.1. *Fix $\gamma < 1$. Then there exists an algorithm running in time* $\mathrm{poly}(n)$ *that, given H, a group of size n, constructs a symmetric set $S \subseteq H$ of size $|S| = O(\frac{\log n}{\gamma^2})$ such that $\lambda(\mathrm{Cay}(H; S)) \leq \gamma$.*

We prove this after presenting the randomized algorithm.

7.5.2 A randomized algorithm

Theorem 7.5.2 ([AR94, LR04, LS04])**.** *Fix $0 < \gamma < 1$, and let H be a group of size n. Identify H with $[n]$. Let $X_1, \ldots, X_k$ be chosen randomly in H, where $k = O(\frac{\log n}{\gamma^2})$. We let the multi-set S be $(X_1, \ldots, X_k)$, and we have*

$$\Pr_{S \subseteq H}[\lambda(\mathrm{Cay}(H; S \sqcup S^{-1})) > \gamma] < 1$$

where $S \sqcup S^{-1}$ denotes the symmetric closure of S, namely the number of occurrences of s and s^{-1} in $S \sqcup S^{-1}$ equals the number of occurrences of s in S, .

To identify the notation in the following proof precisely with that used in Section 7.4, we have that S corresponds to $\vec{X}$, $|S| = k$, and it will become clear that in this setting $n = d = |H|$.

Proof. Consider the $n \times n$ matrices P_h for $h \in H$, where each P_h is the $n \times n$ permutation matrix of the action of h by right multiplication. Consider now $\frac{1}{2}(P_h + P_{h^{-1}})$. It is not hard to see that the normalized adjacency matrix A of $\mathrm{Cay}(H; S \sqcup S^{-1})$ is given by

$$A = \frac{1}{k} \sum_{i=1}^{k} \frac{1}{2}(P_{X_i} + P_{X_i^{-1}})$$

We wish to bound $\lambda(A)$. We know that the largest eigenvalue is 1 and corresponds to J/n where J is the all 1 matrix. Since we want to analyze the second-largest

eigenvalue, we consider

$$(I - J/n)A = \frac{1}{k}\sum_{i=1}^{k}(I - J/n)\frac{1}{2}(P_{X_i} + P_{X_i^{-1}})$$

We let our matrix-valued function be $f(h) = (I - J/n)\frac{1}{2}(P_h + P_{h^{-1}})$, so that

$$\lambda(A) = \|(I - J/n)A\| = \|\frac{1}{k}\sum_{i=1}^{k} f(X_i)\|$$

It is straightforward to verify that $f(h) \in \mathcal{M}_n$, $\|f(h)\| \leq 1$ and $\mathbb{E}_{h\in H}[f(h)] = 0$.

Thus we may apply Theorem 7.2.6 to get that

$$\Pr[\lambda(A) > \gamma] = \Pr[\|\frac{1}{k}\sum_{i=1}^{k} f(X_i)\| > \gamma] \tag{7.5.1}$$

$$\leq 2ne^{-\gamma^2|S|/4} \tag{7.5.2}$$

so picking $k = O(\frac{\log n}{\gamma^2})$ suffices to make this probability less than 1. ■

7.5.3 Derandomizing

Proof of Theorem 7.5.1. To derandomize and obtain Theorem 7.5.1, we apply Corollary 7.4.2 to obtain efficient pessimistic estimators for the event $\sigma(S) = 1$, which holds iff $\|\frac{1}{k}\sum_{i=1}^{k} f(X_i)\| \leq \gamma$. We fix $k = O(\frac{1}{\gamma^2}\log n)$ large enough such that the probability of this event is non-zero (i.e. the estimators we got are useful). We then apply Theorem 7.3.2 to greedily choose successive elements of H to be put in S in order to make an expander. ■

7.5.4 Derandomized Homomorphism Testing

Theorem 7.5.1 answers a question about the derandomization of homomorphism testers posed by Shpilka and Wigderson [SW04]. In this section we will use Theorem 7.5.1 to prove Corollary 7.5.4.

An *affine homomorphism* between two groups H, H' is a map $f : H \to H'$ such that $f^{-1}(0)f$ is a homomorphism. An (δ, η)-test for affine homomorphisms is a tester that accepts any affine homomorphism surely and rejects with probability $1 - \delta$ any $f : H \to H'$ which is η far from being an affine homomorphism. Here distance is measured by the normalized Hamming distance: $d(f, g) = \Pr[f(x) \neq g(x)]$, where the probability is over x chosen uniformly from H.

[SW04] showed how to efficiently construct a tester $T_{H\times S}$ using an expander $\mathrm{Cay}(H; S)$ where $\lambda(\mathrm{Cay}(H; S)) < \lambda$: simply pick a random element $x \overset{R}{\leftarrow} H$ and a random element of $y \overset{R}{\leftarrow} S$ and check to see that $f(0)f(x)^{-1}f(xy) = f(y)$. It is clear this accepts f surely if f is an affine homomorphism. [SW04] shows that if $12\delta < 1 - \lambda$ then this rejects with probability $1 - \delta$ any f that is $\frac{4\delta}{1-\lambda}$-far from being an affine homomorphism.

Theorem 7.5.3 ([SW04])**.** *For all groups H, H' and $S \subseteq H$ an expanding generating set such that $\lambda(\mathrm{Cay}(H; S)) < \lambda$, we can construct a tester $T_{H\times S}$ that surely accepts any affine homomorphism $f : H \to H'$ and rejects with probability at least $1 - \delta$ any $f : H \to H'$ which is $4\delta/(1 - \lambda)$ far from being an affine homomorphism, given that $\frac{12\delta}{1-\lambda} < 1$. That is, $T_{H\times S}$ is a $(\delta, \frac{4\delta}{1-\lambda})$-test for affine homomorphisms.*

In [SW04] the deterministic construction of S gave a set of size $|H|^{\varepsilon}$ for arbitrary $\varepsilon > 0$. The explicit construction given in [SW04] requires that $T_{H\times S}$ use $(1 + \varepsilon) \log |H|$ random bits and asks whether it is possible to improve this dependency on randomness. Theorem 7.5.1 allows us indeed to improve this dependency to the following.

Corollary 7.5.4. *Given an arbitrary group H, one can construct in time $|H|^{O(1)}$ a* homomorphism tester *for functions on H which uses only $\log |H| + \log \log |H| + O(1)$ random bits.*

Proof of Corollary 7.5.4. Theorem 7.5.3 says we can construct a homomorphism tester that only uses randomness to pick an element of H and an element of an expanding generating set of H. Theorem 7.5.1 implies this only requires $\log|H| + \log\log|H| + O(1)$ random bits since we can deterministically construct an expanding generating set of size $\log|H|$ in polynomial time. ■

7.6 Covering SDP's

Linear programming (LP) was one of the first tools computer scientists used to approximate **NP**-hard problems. As a natural relaxation of integer programming (IP), linear programs give fractional solutions to an IP, which may then be rounded to give provably good solutions to the original IP.

More recently, a more general class of relaxations, *semi-definite programs* (SDP's), have been used by computer scientists (e.g. [GW95, ARV04]) to give better approximation guarantees to **NP**-hard problems. SDP's may be solved in polynomial time (using e.g. the ellipsoid method or interior-point methods, see [Sho77, Sho87, YN77, VB96]), and again the solution may be rounded to give a solution to the original IP.

In this section we will define a restricted class of integer SDP's and show that our pessimistic estimators will give a good approximation guarantee.

7.6.1 Definition

We define the notion of *integer covering SDP's*, which are generalizations of integer covering linear programs (see e.g. [KY05]). These programs take the following form:

given $c \in [0,1]^n$ and $f : [n] \to [0, I_d]$,[2] find $y \in \mathbb{N}^n$ where

$$\text{minimize } c^T y \tag{7.6.1}$$

$$\text{with feasibility constraint } y_1 f(1) + \ldots y_n f(n) \geq I \tag{7.6.2}$$

where the feasibility inequality uses the p.s.d. ordering. The vector c may be interpreted as a cost vector, and we wish to minimize the cost of a solution $y \in \mathbb{N}^n$. This is relaxed into a covering SDP by allowing $y \in \mathbb{R}_+^n$ where $\mathbb{R}_+$ denotes the non-negative reals, which we would then like to round y to a solution $\hat{y} \in \mathbb{N}^n$ that is not too much more costly. We will let OPT denote the optimal value of the *relaxed* covering SDP.

Our main theorem is as follows:

Theorem 7.6.1. *Suppose we have a program as in Equation 7.6.1 and suppose we have a feasible relaxed solution vector $y \in \mathbb{R}_+^n$. Then we can find in time* $\text{poly}(n, d)$ *a feasible integer solution $\hat{y}$ such that*

$$c^T \hat{y} \leq O(\log d) \cdot c^T y$$

Corollary 7.6.2. *Given an integer covering SDP with optimum OPT, we can efficiently find an integer solution with cost at most $O(\log d) \cdot OPT$.*

This is done by using a randomized rounding algorithm given implicitly in [AW02], and then derandomizing using pessimistic estimators.

Also, note that this is a natural generalization of integer covering linear programs of the following form: for a cost vector $c \in \mathbb{R}_+^n$, a matrix $A \in \mathbb{R}_+^{d \times n}$

$$\text{minimize } c^T y$$

$$\text{subject to feasibility constraints that for all } i \in [d]\text{: } (Ay)_i \geq 1$$

[2] We restrict ourself to this scale for simplicity. Our results apply to any bounded function with a constant loss in efficiency.

This may be viewed as the special case of integer covering SDP's where all the matrices are diagonal; each $f(i)$ is just the diagonal matrix with i'th column of A along the diagonal. Integer covering LP's, in turn, are a generalization of the very familiar set cover problem, which are exactly the programs where the columns of A are either 0 or 1. In the language of set cover, the universe is $[n]$ and the columns of A are the indicator vectors for the sets we may use to cover $[n]$.

Our approximation for integer covering SDP's will imply a new approximation algorithm for all these covering problems with a logarithmic approximation guarantee. Thus in a sense our algorithm gives optimal approximation factors (up to constants), since a logarithmic approximation factor is optimal (up to constant factors) assuming that $\mathbf{P} \neq \mathbf{NP}$, as shown by [Fei98]. This connection is discussed in more detail in Section 7.6.4.

7.6.2 A randomized rounding algorithm

First suppose we have a solution to the SDP given by a vector $y \in \mathbb{R}^n_+$, and let us define $Q = \sum_{j=}^{n} y_j$. In the case where $Q \geq n$, we can get a trivial deterministic rounding scheme with approximation factor 2 by always rounding up, since this will increase the value of the program at most by an additive n. Thus in the following we consider only programs where $Q \leq n$.

Suppose we have a program as in Equation 7.6.1 and we have solved it efficiently to obtain a solution y, where $c^T y = OPT$. Let X be distributed according to the distribution over $[n]$ given by normalizing y, i.e.

$$\Pr[X = i] = y_i/Q$$

Note that, because y is a feasible solution, we have $\mathbb{E}_X[f(X)] \geq \frac{1}{Q} I$. It was implicitly shown in [AW02] that sampling $k = Q \cdot O(\log d)$ elements from $[n]$ according to

the distribution X and taking $f(X_i)$ $(1 \leq i \leq k)$ gives us a feasible solution with approximation factor $O(\log d)$. We state this formally:

Theorem 7.6.3. *[[AW02]] Suppose we sample $k = Q \cdot 8 \ln 2d$ times from $[n]$ according to X in order to get $X_1, \ldots, X_k$. Furthermore, for each $1 \leq j \leq n$, we define the random variables*

$$\hat{Y}_j = |\{i \mid X_i = j\}|$$

the number of times that j is sampled, and let $\hat{Y} = (\hat{Y}_1, \ldots, \hat{Y}_n)$. Notice that $\sum_{i=1}^{k} f(X_i) = \sum_{j=1}^{n} \hat{Y}_j f(j)$. Then, with non-zero probability, we have that

$$f(X_1) + f(X_2) + \ldots + f(X_k) \geq I \qquad \text{and} \qquad c^T \hat{Y} \leq c^T y \cdot 16 \ln 2d$$

Proof. We will use a union bound to show that the probability that either $\sum_j f(X_j) \not\geq I$ or $c^T \hat{Y} > c^T y \cdot 16 \ln 2d$ occurs is strictly less than 1.

All expectations below are over the X_i (since the $\hat{Y}_j$ are totally determined by the X_i).

$$\Pr[\sum_{j=1}^{k} f(X_j) \not\geq I] = \Pr[\tfrac{1}{k} \sum_{j=1}^{k} f(X_j) \not\geq \tfrac{1}{k} I] \tag{7.6.3}$$

We know from the fact that y is feasible that $\mathbb{E}[f(X)] \geq \frac{1}{Q} I$, and so for $k > 2Q$ we get:

$$\Pr[\sum_{j=1}^{k} f(X_j) \not\geq I] \leq \Pr[\tfrac{1}{k} \sum_{j=1}^{k} f(X_j) \not\geq \tfrac{1}{2} \tfrac{1}{Q} I] \tag{7.6.4}$$

Invoking Theorem 7.2.8, we obtain

$$\Pr[\sum_{j=1}^{k} f(X_j) \not\geq I] \leq d e^{\frac{-k}{(8Q)}} \tag{7.6.5}$$

Therefore if we take $k = Q \cdot 8 \ln 2d$ with probability greater than $\frac{1}{2}$ we have $\sum_j f(X_j) \geq I$.

For the second event it is easy to see that $c^T\hat{Y} = \sum_{j=1}^{k} c_{X_j}$. Furthermore, a simple calculation shows that for each j, $\mathbb{E}[c_{X_j}] = c^T y/Q$. Thus, by Markov we have:

$$\Pr[c^T\hat{Y} > c^T y \cdot 16 \ln 2d] = \Pr\left[\sum_{j=1}^{k} c_{X_j} > c^T y \cdot 16 \ln 2d\right] \tag{7.6.6}$$

$$< \frac{\mathbb{E}\left[\sum_{j=1}^{k} c_{X_j}\right]}{c^T y \cdot 16 \ln 2d} \tag{7.6.7}$$

$$= \frac{k \cdot c^T y/Q}{c^T y \cdot 16 \ln 2d} \tag{7.6.8}$$

Expanding $k = Q \cdot 8 \ln 2d$ shows that this last expression is at most $1/2$.

Thus each bad event happens with probability less than $1/2$, and so the probability that either bad event happens is strictly less than 1. ■

7.6.3 Derandomizing

Derandomizing is a simple proposition. Given a program, first solve it using a standard efficient technique ([Sho77, Sho87, YN77], for a survey see [VB96]), with solution y and $Q = \sum_{j=1}^{n} y_j$. Let $k = Q \cdot 8 \ln 2d$. In the proof of Theorem 7.6.3 at Inequality 7.6.3, we can apply Theorem 7.4.4 to get pessimistic estimators ϕ_i for the event $\sum_{j=1}^{k} f(X_j) \geq I$, which we call σ. We only need now a pessimistic estimator $(\psi_0, \ldots, \psi_k)$ for the event of the solution not being too costly, which we call τ.

We define $\psi_i : [n]^i \to [0, 1]$ as follows:

$$\psi_i(x_1, \ldots, x_i) = \frac{\sum_{j=1}^{i} c_{x_j} + (k - i)\mathbb{E}[c_X]}{c^T y \cdot 16 \ln 2d}$$

It is clear that the ψ_i are efficiently computable. They satisfy the properties of Definition 7.3.1. This is easy to see, since the ψ_i are exactly the expressions given by a Markov bound on the event τ, and such expressions always satisfy Definition 7.3.1. We write this out explicitly here fore completeness.

1. By an application of Markov (this is the same as in Inequality 7.6.7), we see:

$$\Pr\left[\sum_{j=1}^{k} c_{X_j} > c^T y \cdot 16 \ln 2d \mid X_1 = x_1, \ldots, X_i = x_i\right] \leq \frac{\sum_{j=1}^{i} c_{x_j} + (k-i)\mathbb{E}[c_X]}{c^T y \cdot 16 \ln 2d}$$
$$= \psi(x_1, \ldots, x_i)$$

2. For estimators based on Markov, we actually have equality for this property.

$$\mathbb{E}_{X_{i+1}}[\psi_{i+1}(x_1, \ldots, x_i, X_{i+1})] = \mathbb{E}_{X_{i+1}}\left[\frac{\sum_{j=1}^{i} c_{x_j} + c_{X_{i+1}} + (k-i-1)\mathbb{E}[c_X]}{c^T y \cdot 16 \ln 2d}\right]$$
$$= \frac{\sum_{j=1}^{i} c_{x_j} + (k-i)\mathbb{E}[c_{X_j}]}{c^T y \cdot 16 \ln 2d}$$
$$= \psi_i(x_1, \ldots, x_i)$$

Theorem 7.6.4. *Since $\phi_0 + \psi_0 < 1$ because of the choice of $k = Q \cdot 8 \ln 2d$, we may invoke Lemma 7.3.3 to get that $(\phi_0 + \psi_0, \ldots, \phi_k + \psi_k)$ are efficient and useful pessimistic estimators for the event in Theorem 7.6.3.*

Finally we may prove Theorem 7.6.1.

Proof of Theorem 7.6.1. By Theorem 7.6.4 we have pessimistic estimators for the event in Theorem 7.6.3, and so we may apply Theorem 7.3.2, which says we can efficiently and deterministically find a suitable integer vector $\hat{y}$ that satisfies Theorem 7.6.1. The algorithm runs in time $\text{poly}(n, k, d)$, but since $k = Q \cdot 8 \ln 2d$ and we only consider $Q \leq n$, this is $\text{poly}(n, d)$. ■

7.6.4 Quantum Hypergraph Covers

In this section we define hypergraphs and quantum hypergraphs and discuss the cover problem for both. The hypergraph cover problem is just the classical set cover

problem, and the quantum hypergraph cover problem is a non-commutative generalization arising in quantum information theory [AW02]. Our efficient and useful pessimistic estimators for the integer covering SDP problem immediately give an efficient deterministic algorithm to find a quantum hypergraph cover that is optimal up to logarithmic factors.

Hypergraphs

Here we will describe the hypergraph cover problem, which is just another name for the classical set cover. A hypergraph is a pair (V, E) where $E \subseteq 2^V$, i.e. E is a collection of subsets of V. Say $|V| = d$. One often views an edge e as a vector in $\{0,1\}^d$, where the i'th entry is 1 if vertex i is in the edge and 0 otherwise.

It will actually be convenient for us to view $e \in E$ as $d \times d$ diagonal matrix with 1 or 0 at each diagonal entry to signify whether that vertex is in the edge. In this section we will denote the matrix associated with e as $f(e)$. This representation will naturally generalize to quantum hypergraphs.

A *cover* of a hypergraph $\Gamma = (V, E)$ is a set of edges C such that $\bigcup_{e \in C} e = V$, i.e. each vertex is in at least one edge. Note that this definition of cover coincides exactly with the definition of set cover. The size of the smallest cover is called the *cover number* and dentoted $c(\Gamma)$.

Using the matrix representation of E, one sees that

$$\bigcup_{e \in C} e = V \quad \Leftrightarrow \quad \sum_{e \in C} f(e) \geq I$$

where the second expression uses our usual ordering of matrices.

A *fractional cover* is a set of non-negative weights w over E such that $\sum_{e \in E} w(e) f(e) \geq$

I. Likewise, we say that the *fractional cover number*

$$\tilde{c}(\Gamma) = \min_{w} \left\{ \sum_{e \in E} w(e) \middle| \sum_{e \in E} w(e) f(e) \geq I \right\}$$

We know that the hypergraph cover problem is hard to approximate up to a $\ln n$ factor [Fei98]. From the definitions, it is clear that this problem is a special case of our integer covering SDP's. In the next section we generalize to the non-commutative case.

Quantum Hypergraphs

[AW02] defines *quantum hypergraphs* as generalizations of hypergraphs. Recall that we represented an edge of a hypergraph as a $d \times d$ diagonal matrix with $1, 0$ along the diagonal. So a hypergraph is equivalent to $(\mathcal{V}, \mathcal{E})$ where $\mathcal{V} = \mathbb{C}^d$ and each $e \in \mathcal{E}$ is identified with a diagonal matrix whose diagonal entries are either 0 or 1, which we will call $f(e)$. We generalize this to non-commutative "edges" by allowing $\mathcal{E}$ to contain other operators, i.e. $f(e)$ can be any Hermitian operator (i.e. matrix) in $[0, I]$. Here we are using the fact that all our previous results for real symmetric matrices generalize to complex Hermitian matrices. A complex matrix A is Hermitian if $A = A^*$ where * denotes the conjugate transpose. See Section 7.7 for a statement of our results in this (and a slightly more general) setting.

Definition 7.6.5. $\Gamma = (\mathcal{V}, \mathcal{E})$ is a quantum hypergraph where $\mathcal{V}$ is a d-dimensional Hilbert space and $\mathcal{E}$ is a finite set such that each $e \in \mathcal{E}$ is identified with a Hermitian operator $f(e) \in [0, I_d]$.

One can extend the definition of a cover of a quantum hypergraph $\Gamma = (\mathcal{V}, \mathcal{E})$ to be a finite subset $C \subseteq \mathcal{E}$ such that $\sum_{e \in C} f(e) \geq I$. The cover number $c(\Gamma)$ is the size of the smallest cover of Γ.

Likewise, we define a fractional cover to be a non-negative combination w of $e \in \mathcal{E}$ such that $\sum_{e\in\mathcal{E}} w(e)f(e) \geq I$, and the fractional cover number as

$$\tilde{c}(\Gamma) = \min_{w} \left\{ \sum_{e\in\mathcal{E}} w(e) \,\middle|\, \sum_{e\in\mathcal{E}} w(e)f(e) \geq I \right\}$$

Note that this corresponds exactly with our previous definitions for hypergraphs. The problem of finding the fractional cover has equivalent forms that are natural and interesting, which are discussed at the end of this section.

It is important to note that the notion of "vertex" is lost because the matrices $f(e) \in \mathcal{M}_d$ are not necessarily diagonal in a common basis. However, it is again clear from the definitions that a quantum hypergraph cover problem is just a special case of integer covering SDP's (extended to complex matrices), so we may use Theorem 7.6.1 to give an efficient deterministic approximation. Thus the theorem below follows.

Theorem 7.6.6. *Suppose we are given $\Gamma = (\mathcal{V}, \mathcal{E})$ a quantum hypergraph with fractional cover number $\tilde{c}(\Gamma)$, with $|\mathcal{V}| = d$ and $|\mathcal{E}| = n$. Then we can find an integer cover of Γ of size $k = \tilde{c}(\Gamma) \cdot O(\log d)$ in time* $\mathrm{poly}(n, d)$.

7.6.5 Other Applications

Our integer covering SDP (and its extension to complex matrices) also encompasses two other natural problems from quantum information theory. Given a function $f : [n] \to [0, I_d]$, one may want to find a probability distribution X over $[n]$ one may want to solve either of the following

1. $\min_X \lambda_1(\mathbb{E}_X[f(X)]) = \min_X \|\mathbb{E}_X[f(X)]\|$
2. $\max_X \lambda_d(\mathbb{E}_X[f(X)])$

The former minimizes the norm of the expected value of the distribution, which is also its largest eigenvalue, while the latter may be viewed as maximizing the lowest energy state of a quantum system, which is also its smallest eigenvalue. The second can be formulated as a covering SDP by using the cost vector $c = \mathbf{1}$ the all 1's vector, and then normalizing the solution vector y to be a probability distribution. The first can be formulated as the second by considering the function $I - f$.

In both cases, our pessimistic estimators give an "integral solution" that is worse by at most $O(\log d)$. In this case, an integral solution is actually a distribution with sparse support; we sample from the solution distribution X to get a distribution $\hat{X}$ with support of size $O(\frac{1}{\gamma^2} \log d)$ such that the corresponding objective is worse by at most a factor of $O(\log d)$.

7.7 Generalization to abstract vector spaces

Here we state the generalization of all of our results to the setting of self-adjoint operators over Hilbert space.

All our theorems from Section 7.4 hold in the setting of abstract finite-dimensional Hilbert spaces as stated, with the linear-algebra terminology translated to the terminology of finite-dimensional Hilbert spaces. Table 7.1 gives this translation. The proofs of the generalizations are identical to the proofs of the real symmetric case with the appropriate change in terminology.

Using these correspondences, the definitions of trace, positive semi-definiteness, the p.s.d. ordering, and the exponential on $\mathcal{L}(V)$ carry over from the real matrix case (given in Section 7.2) in the natural way. For example, the notation $[-I, I]$ indicates all self-adjoint operators $A \in \mathcal{L}(V)$ such that the eigenvalues of $A + I$ and $I - A$ are

Real symmetric matrices	Abstract Hilbert space
$\mathbb{R}^d$	V Hilbert space of finite dimension d
Inner product $\langle \cdot, \cdot \rangle$ over $\mathbb{R}^d$	Inner product $\langle \cdot, \cdot \rangle$ over V
Norm $\| \cdot \|$ over $\mathbb{R}^d$	$\|v\| = \langle v, v \rangle^{1/2}$
$d \times d$ matrices	$\mathcal{L}(V)$ = linear operators from V to itself
$\mathcal{M}_d$ = symmetric $d \times d$ matrices	$\mathcal{S}_V$ = self-adjoint operators in $\mathcal{L}(V)$: $A \in \mathcal{L}(V)$ satisfying for all $v, w \in V$ the identity $\langle v, Aw \rangle = \langle Av, w \rangle$.
Matrix norm $\|A\| = \max_v \|Av\|/\|v\|$	Operator norm $\|A\| = \max_v \|Av\|/\|v\|$

Table 7.1: Translation of terminology to abstract Hilbert space setting.

non-negative.

Ahlswede and Winter [AW02] already stated all their results in this setting an abstract Hilbert spaces, and so Theorem 7.2.4, Theorem 7.2.6, and Theorem 7.2.8 are all valid with $\mathcal{M}_d$ replaced by $\mathcal{S}_V$. Notice that we have set the dimension of V to the same as the dimension of $\mathbb{R}^d$, which is why the same bounds hold verbatim in both the real symmetric and abstract Hilbert space cases.

Similarly, all of our results from Section 7.4 hold with $\mathcal{M}_d$ replaced by S_V, *i.e.* considering functions f with image in $\mathcal{S}_V$ instead of in $\mathcal{M}_d$. The sole caveat here is that in order for the estimators to be efficient, we need addition, multiplication, trace, exponential, and norm all to be efficiently computable for the operators in $\mathcal{S}_V$.

Bibliography

[AKS02] M. Agrawal, N. Kayal, and N. Saxena. PRIMES is in P. *Ann. of Math*, 2:781–793, 2002.

[AW02] R. Ahlswede and A. Winter. Strong Converse for Identification via Quantum Channels. *IEEE Trans. Inf. The.*, 48(3):569–579, 2002.

[AH91] W. Aiello and J. Hastad. Statistical Zero-Knowledge Languages can be Recognized in Two Rounds. *JCSS*, 42:327–345, 1991.

[Ajt96] M. Ajtai. Generating hard instances of lattice problems (extended abstract). In *Proc. 28th STOC*, pages 99–108, New York, NY, USA, 1996. ACM.

[AGGM06] A. Akavia, O. Goldreich, S. Goldwasser, and D. Moshkovitz. On basing one-way functions on NP-hardness. In *Proc. 38th STOC*, pages 701–710, New York, NY, USA, 2006. ACM.

[ABF+04] M. Alekhnovich, M. Braverman, V. Feldman, A. R. Klivans, and T. Pitassi. Learnability and Automatizability. In *FOCS '04*, pages 621–630, 2004.

[AFWZ95] N. Alon, U. Feige, A. Wigderson, and D. Zuckerman. Derandomized Graph Products. *Computational Complexity*, 5(1):60–75, 1995.

[AR94] N. Alon and Y. Roichman. Random Cayley Graphs and Expanders. *RSA: Random Structures & Algorithms*, 5, 1994.

[ABX08] 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.

[AMCS04] K. Argyraki, P. Maniatis, D. Cheriton, and S. Shenker. Providing Packet Obituaries. *ACM HotNets-III*, 2004.

[AB09] S. Arora and B. Barak. *Complexity Theory: A Modern Approach*. Cambridge University Press, 2009.

[AK07] S. Arora and S. Kale. A combinatorial, primal-dual approach to semidefinite programs. In *Proc. 39th STOC*, pages 227–236, New York, NY, USA, 2007. ACM.

[ALM+98] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof Verification and the Hardness of Approximation Problems. *Journal of the ACM*, 45(3):501–555, May 1998.

[ARV04] S. Arora, S. Rao, and U. Vazirani. Expander flows, geometric embeddings, and graph partitionings. In *Proc. 36th STOC*, pages 222–231. ACM, 2004.

[AKWK04] I. Avramopoulos, H. Kobayashi, R. Wang, and A. Krishnamurthy. Highly secure and efficient routing. *INFOCOM 2004*, 1:208, 7-11 March 2004.

[AHNRR02] B. Awerbuch, D. Holmer, C. Nita-Rotaru, and H. Rubens. An secure routing protocol resilient to byzantine failures. In *WiSE '02*, pages 21–30. ACM, 2002.

[BGS75] T. Baker, J. Gill, and R. Solovay. Relativizations of the $\mathcal{P}$ =?$\mathcal{NP}$ Question. *SIAM Journal on Computing*, 4(4):431–442, 1975.

[BGX08] B. Barak, S. Goldberg, and D. Xiao. Protocols and Lower Bounds for Secure Failure Localization on the Internet. In *Proc. EUROCRYPT '08*, pages 341–360, 2008.

[BS84] J. Beck and J. Spencer. Integral Approximatino Sequences. *Mathematical Programming*, 30(1):88–98, 1984.

[BDEL03] S. Ben-David, N. Eiron, and P. M. Long. On the difficulty of approximately maximizing agreements. *J. Comput. Syst. Sci.*, 66(3):496–514, 2003.

[BOGG+88] M. Ben-Or, O. Goldreich, S. Goldwasser, J. Håstad, J. Kilian, S. Micali, and P. Rogaway. Everything Provable is Provable in Zero-Knowledge. In *Proc. 8th CRYPTO*, pages 37–56, London, UK, 1988. Springer-Verlag.

[BH04] Y. Bilu and S. Hoory. Hypergraph Codes. *European Journal of Combinatorics*, 25(3):339–354, 2004.

[BFKL93] A. Blum, M. L. Furst, M. J. Kearns, and R. J. Lipton. Cryptographic Primitives Based on Hard Learning Problems. In *CRYPTO '93*, pages 278–291, 1993.

[BR92] A. L. Blum and R. L. Rivest. Training a 3-node neural network is NP-complete. *Neural Netw.*, 5(1):117–127, 1992.

[BLR90] M. Blum, M. Luby, and R. Rubinfeld. Self-Testing/Correcting with Applications to Numerical Problems. *Journal of Computer and System Sciences*, 47:549–595, 1990.

[BM84] M. Blum and S. Micali. How to generate cryptographically strong sequences of pseudo-random bits. *SIAM J. Comput.*, 13(4):850–864, 1984.

[BT06] A. Bogdanov and L. Trevisan. On Worst-Case to Average-Case Reductions for NP Problems. *SIAM J. Comput.*, 36(4):1119–1159, 2006.

[BHZ87] R. B. Boppana, J. Hastad, and S. Zachos. Does co-NP have short interactive proofs? *Inf. Process. Lett.*, 25(2):127–132, 1987.

[CW89] A. Cohen and A. Wigderson. Dispersers, Deterministic Amplification, and Weak Random Sources. In *Proc.* 30*th FOCS*, pages 14–19. IEEE, 1989.

[CK05] M. Collins and T. Koo. Discriminative Reranking for Natural Language Parsing. *Comput. Linguist.*, 31(1):25–70, 2005.

[CK06] M. Crovella and B. Krishnamurthy. *Internet Measurement*. Wiley, 2006.

[Dam93] I. Damgård. Interactive Hashing can Simplify Zero-Knowledge Protocol Design Without Computational Assumptions (Extended Abstract). In *Proc.* 13*th CRYPTO*, pages 100–109, London, UK, 1993. Springer-Verlag.

[DGOW95] I. Damgård, O. Goldreich, T. Okamoto, and A. Wigderson. Honest Verifier vs Dishonest Verifier in Public Cain Zero-Knowledge Proofs. In *Proc. 15th CRYPTO*, pages 325–338, London, UK, 1995. Springer-Verlag.

[DSDCPY98] A. De Santis, G. Di Crescenzo, G. Persiano, and M. Yung. Image Density is Complete for Non-Interactive-SZK (Extended Abstract). In *Proc. 25th ICALP*, pages 784–795, London, UK, 1998. Springer-Verlag.

[DDWY93] D. Dolev, C. Dwork, O. Waarts, and M. Yung. Perfectly secure message transmission. *J. ACM*, 40(1):17–47, 1993.

[DSS93] H. Drucker, R. E. Schapire, and P. Simard. Boosting Performance in Neural Networks. *IJPRAI*, 7(4):705–719, 1993.

[DG01] N. G. Duffield and M. Grossglauser. Trajectory sampling for direct traffic observation. *IEEE/ACM Trans. Netw.*, 9(3):280–292, 2001.

[ESY84] 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.

[Fei98] U. Feige. A Threshold of $\ln n$ for Approximating Set Cover. *Journal of the ACM*, 45(4):634–652, 1998.

[FF93] J. Feigenbaum and L. Fortnow. Random-self-reducibility of complete sets. *SIAM J. Comput.*, 22(5):994–1005, 1993.

[Fel06a] V. Feldman. Hardness of approximate two-level logic minimization and PAC learning with membership queries. In *STOC '06*, pages 363–372, 2006.

[Fel06b] V. Feldman. Optimal Hardness Results for Maximizing Agreements with Monomials. *CCC '06*, pages 226–236, 2006.

[For87] L. Fortnow. The complexity of perfect zero-knowledge. In *STOC '87*, pages 204–209, 1987.

[Fre90] Y. Freund. Boosting a weak learning algorithm by majority. In *Proc. COLT '90*, pages 202–216, San Francisco, CA, USA, 1990. Morgan Kaufmann Publishers Inc.

[FS97] Y. Freund and R. E. Schapire. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting,. *Journal of Comp. and Sys. Sci.*, 55(1):119–139, 1997.

[GGK03] R. Gennaro, Y. Gertner, and J. Katz. Lower bounds on the efficiency of encryption and digital signature schemes. In *Proc.* 35*th STOC*, pages 417–425, New York, NY, USA, 2003. ACM.

[GT00] R. Gennaro and L. Trevisan. Lower bounds on the efficiency of generic cryptographic constructions. In *Proc.* 41*st FOCS*, pages 305–313. IEEE, 2000.

[GW95] M. X. Goemans and D. P. Williams. Improved approximation algorithms for Max-Cut and Satisfiability Problems using Semidefinite Programming. *J. of the ACM*, 42:1115–1145, 1995.

[GXB+08] S. Goldberg, D. Xiao, B. Barak, J. Rexford, and E. Tromer. Path-Quality Monitoring in the Presence of Adversaries. In *ACM SIGMETRICS*, 2008.

[Gol65] S. Golden. Lower Bounds for the Helmholtz Function. *Physical Review*, 137B(4):B1127–1128, 1965.

[Gol97] O. Goldreich. A Sample of Samplers - A Computational Perspective on Sampling (survey). *Electronic Colloquium on Computational Complexity (ECCC)*, 4(020), 1997.

[GGM86] O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions. *Journal of the ACM*, 33(4):792–807, 1986. Preliminary version in FOCS' 84.

[GGR98] O. Goldreich, S. Goldwasser, and D. Ron. Property testing and its connection to learning and approximation. *J. ACM*, 45(4):653–750, 1998.

[GIL+90] O. Goldreich, R. Impagliazzo, L. Levin, R. Venkatesan, and D. Zuckerman. Security Preserving Amplification of Hardness. In *Proc.* 31*st FOCS*, pages 318–326. IEEE, 1990.

[GMW86] O. Goldreich, S. Micali, and A. Wigderson. Proofs that Yield Nothing But Their Validity or All Languages in NP Have Zero-Knowledge Proof Systems. *Journal of the ACM*, 38(3):691–729, July 1991. Preliminary version in FOCS' 86.

[GSV98] O. Goldreich, A. Sahai, and S. Vadhan. Honest-verifier statistical zero-knowledge equals general statistical zero-knowledge. In *Proc. STOC '98*, pages 399–408, New York, NY, USA, 1998. ACM.

[GS02] O. Goldreich and M. Sudan. Locally testable codes and PCPs of almost-linear length. In *Proc.* 43*rd FOCS*, pages 13–22. IEEE Computer Society Press, 2002.

[GV99] O. Goldreich and S. Vadhan. Comparing Entropies in Statistical Zero Knowledge with Applications to the Structure of SZK. In *COCO '99*, pages 54–73, 1999.

[GVW01] O. Goldreich, S. Vadhan, and A. Wigderson. On Interactive Proofs with a Laconic Prover. In *Proc. 28th ICALP*, 2001.

[GMR85] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM J. of Com.*, 18(1):186–208, 1989. Preliminary version in STOC' 85.

[GS89] 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.

[GL89] G. H. Golub and C. F. V. Loan. *Matrix Computations*. Johns Hopkins University Press, 1989.

[GKS07] P. Gopalan, S. Khot, and R. Saket. Hardness of Reconstructing Multivariate Polynomials over Finite Fields. In *FOCS '07*, pages 349–359, 2007.

[GR06] V. Guruswami and P. Raghavendra. Hardness of Learning Halfspaces with Noise. In *FOCS '06*, pages 543–552, 2006.

[HHRS07] 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 *Proc. FOCS '07*, pages 669–679, 2007.

[HR07] I. Haitner and O. Reingold. Statistically-hiding commitment from any one-way function. In *STOC*, pages 1–10, 2007.

[HJLT96] T. Hancock, T. Jiang, M. Li, and J. Tromp. Lower bounds on learning decision lists and trees. *Inf. Comput.*, 126(2):114–122, 1996.

[HILL89] J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby. A pseudo-random generator from any one-way function. *SIAM J. of Com.*, 28(4):1364–1396, 1999. Preliminary versions appeared in STOC' 89 and STOC' 90.

[HR08] J. He and J. Rexford. Towards Internet-wide Multipath Routing. *IEEE Network Magazine Special Issue on Scalablity*, March 2008.

[HLW06] S. Hoory, N. Linial, and A. Wigderson. Expander Graphs and their Applications. *Bull. Amer. Math. Soc.*, 43:439–561, 2006.

[Imp95] R. Impagliazzo. A personal view of average-case complexity. In *SCT '95*, page 134, 1995.

[IL90] R. Impagliazzo and L. A. Levin. No Better Ways to Generate Hard NP Instances than Picking Uniformly at Random. In *FOCS '90*, pages 812–821, 1990.

[IL89] R. Impagliazzo and M. Luby. One-way Functions are Essential for Complexity Based Cryptography (Extended Abstract). In *Proc.* 30*th FOCS*, pages 230–235, 1989.

[IR89] R. Impagliazzo and S. Rudich. Limits on the provable consequences of one-way permutations. In *STOC '89*, pages 44–61. ACM, 1989.

[IW97] R. Impagliazzo and A. Wigderson. **P** = **BPP** if **E** Requires Exponential Circuits: Derandomizing the XOR Lemma. In *Proc. 29th STOC*, pages 220–229. ACM, 1997.

[IW98] R. Impagliazzo and A. Wigderson. Randomness vs. Time: De-Randomization under a Uniform Assumption. In *FOCS*, pages 734–743, 1998.

[IZ89] R. Impagliazzo and D. Zuckerman. How to Recycle Random Bits. In *Proc. 30th FOCS*, pages 248–253. IEEE, 1989.

[Jac88] V. Jacobson. Congestion Avoidance and Control. In *ACM SIGCOMM '88*, pages 314–329, Stanford, CA, Aug. 1988.

[Kar72] R. M. Karp. Reducibility Among Combinatorial Problems. In R. E. Miller and J. W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972.

[KV89] M. Kearns and L. Valiant. Cryptographic limitations on learning Boolean formulae and finite automata. In *Proc. STOC '89*, pages 433–444, New York, NY, USA, 1989. ACM.

[KSS92] M. J. Kearns, R. E. Schapire, and L. M. Sellie. Toward efficient agnostic learning. In *COLT '92*, pages 341–352, 1992.

[KLS00] S. Kent, C. Lynn, and K. Seo. Secure Border Gateway Protocol (S-BGP). *IEEE Jour. Sel. Areas in Comm.*, 18(4):582–592, April 2000.

[KST99] J. H. Kim, D. R. Simon, and P. Tetali. Limits on the Efficiency of One-Way Permutation-Based Hash Functions. In *Proc. 40th FOCS*, page 535, Washington, DC, USA, 1999. IEEE Computer Society.

[KY05] S. G. Kolliopoulos and N. E. Young. Approximation algorithms for covering/packing integer programs. *J. Comput. Syst. Sci.*, 71(4):495–505, 2005.

[LR04] Z. Landau and A. Russell. Random Cayley graphs are expanders: a simplified proof of the Alon-Roichman theorem. *The Electronic Journal of Combinatorics*, 11(2), 2004.

[LC06] P. Laskowski and J. Chuang. Network monitors and contracting systems: competition and innovation. In *SIGCOMM '06*, pages 183–194. ACM, 2006.

[LS04] P.-S. Loh and L. J. Schulman. Improved Expansion of Random Cayley Graphs. *Discrete Mathematics and Theoretical Computer Science*, 6(2):523–528, 2004.

[MSWA03] R. Mahajan, N. Spring, D. Wetherall, and T. Anderson. User-level internet path diagnosis. *SIGOPS Oper. Syst. Rev.*, 37(5):106–119, 2003.

[Mal08] T. Malkin. Personal communication., 2008.

[MFLS01] S. Merler, C. Furlanello, B. Larcher, and A. Sboner. Tuning Cost-Sensitive Boosting and Its Application to Melanoma Diagnosis. In *Proc. MCS '01*, pages 32–42, London, UK, 2001. Springer-Verlag.

[MR04] 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.

[Mil75] G. L. Miller. Riemann's Hypothesis and tests for primality. In *Proc. 7'th STOC*, pages 234–239, New York, NY, USA, 1975. ACM.

[MCMS05] A. Mizrak, Y.-C. Cheng, K. Marzullo, and S. Savage. Fatih: detecting and isolating malicious routers. *DSN 2005*, pages 538–547, 28 June-1 July 2005.

[NN93] J. Naor and M. Naor. Small-Bias Probability Spaces: Efficient Constructions and Applications. *SIAM J. of Com.*, 22(4):838–856, Aug. 1993.

[NR06] M. Naor and G. N. Rothblum. Learning to impersonate. In *ICML 2006*, pages 649–656. ACM, 2006.

[NOV06] M.-H. Nguyen, S.-J. Ong, and S. Vadhan. Statistical Zero-Knowledge Arguments for NP from Any One-Way Function. In *FOCS '06*, pages 3–14, 2006.

[NT99] N. Nisan and A. Ta-Shma. Extracting Randomness: A Survey and New Constructions. *J. Comput. Syst. Sci.*, 58(1):148–173, 1999.

[NW88] N. Nisan and A. Wigderson. Hardness vs Randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, Oct. 1994. Preliminary version in FOCS' 88.

[Oka96] T. Okamoto. On relationships between statistical zero-knowledge proofs. In *Proc. 28th STOC*, pages 649–658, New York, NY, USA, 1996. ACM.

[OV07] S. J. Ong and S. P. Vadhan. Zero Knowledge and Soundness Are Symmetric. In *EUROCRYPT '07*, pages 187–209, 2007.

[Ost91] R. Ostrovsky. One-way functions, hard on average problems, and statistical zeroknowledge proofs. In *In Proc. 6th Annual Structure in Complexity Theory Conf.*, pages 133–138, 1991.

[OW93] R. Ostrovsky and A. Wigderson. One-Way Functions are essential for Non-Trivial Zero-Knowledge. In *ISTCS '93*, pages 3–17, 1993.

[PS03] V. N. Padmanabhan and D. R. Simon. Secure traceroute to detect faulty or malicious routing. *SIGCOMM Comput. Commun. Rev.*, 33(1):77–82, 2003.

[Pas06] R. Pass. Parallel Repetition of Zero-Knowledge Proofs and the Possibility of Basing Cryptography on NP-Hardness. In *Proc. 21st CCC*, pages 96–110, 2006.

[PV88] L. Pitt and L. G. Valiant. Computational limitations on learning from examples. *J. ACM*, 35(4):965–984, 1988.

[PW90] L. Pitt and M. K. Warmuth. Prediction-preserving reducibility. *J. Comput. Syst. Sci.*, 41(3):430–467, 1990.

[Rab80] M. O. Rabin. Probabilistic algorithm for testing primality. *J. Number Theory*, 12(1):128–138, 1980.

[Rag88] P. Raghavan. Probabilistic construction of deterministic algorithms: approximating packing integer programs. *J. Comput. Syst. Sci.*, 37(2):130–143, 1988.

[RTV04] O. Reingold, L. Trevisan, and S. Vadhan. Notions of Reducibility Between Cryptographic Primitives. In *Proc. 1st TCC*, pages 1–20, 2004.

[Riv93] R. L. Rivest. Cryptography and machine learning. In *Proc. ASIACRYPT '91*, pages 427–439. Springer, 1993.

[Rom90] J. Rompel. One-way functions are necessary and sufficient for secure signatures. In *Proc. STOC '90*, pages 387–394, New York, NY, USA, 1990. ACM.

[RS96] R. Rubinfeld and M. Sudan. Robust Characterizations of Polynomials withApplications to Program Testing. *SIAM J. Comput.*, 25(2):252–271, 1996.

[SV97] A. Sahai and S. P. Vadhan. A Complete Promise Problem for Statistical Zero-Knowledge. In *Proc. FOCS '97*, pages 448–457, 1997.

[Sch89] R. Schapire. The strength of weak learnability. *Proc. FOCS '89*, pages 28–33, 1989.

[Sha02] R. Shaltiel. Recent developments in extractors. *Bulletin of the European Association for Theoretical Computer Science*, 2002. Available from http://www.wisodm.weizmann.ac.il/~ronens.

[Sho77] N. Z. Shor. Cut-off method with space extension in convex programming problems. *Cybernetics*, 13:94–96, 1977.

[Sho87] N. Z. Shor. Quadratic optimization problems. *Soviet Journal of Circuits and Systems Sciences*, 25:1–11, 1987.

[SW04] A. Shpilka and A. Wigderson. Derandomizing homomorphism testing in general groups. In *Proc. 36th STOC*, pages 427–435. ACM, 2004.

[Sim98] D. R. Simon. Finding collisions on a one-way street: Can secure hash functions be based on general assumptions? In *Proc. EUROCRYPT '98*, volume 1403, pages 334–345, 1998.

[SS77] R. Solovay and V. Strassen. A Fast Monte-Carlo Test for Primality. *SIAM Journal on Computing*, 6(1):84–85, 1977.

[Spe94] J. Spencer. *Ten Lectures on the Probabilistic Method, 2nd Edition.* SIAM, 1994.

[Spi95] D. Spielman. *Computationally Efficient Error-Correcting Codes and Holographic Proofs.* PhD thesis, M.I.T., 1995.

[SRS+04] L. Subramanian, V. Roth, I. Stoica, S. Shenker, and R. H. Katz. Listen and whisper: security for BGP. In *USENIX NSDI 2004*, pages 10–10, 2004.

[Tho65] C. J. Thompson. Inequality with Applications in Statistical Mechanics. *Journal of Mathematical Physics*, 6(11):1812–1823, 1965.

[Vad04] S. P. Vadhan. An Unconditional Study of Computational Zero Knowledge. *FOCS '04*, pages 176–185, 2004.

[Val84] L. G. Valiant. A theory of the learnable. *Commun. ACM*, 27(11):1134–1142, 1984.

[VB96] L. Vandenberghe and S. Boyd. Semidefinite Programming. *SIAM Review*, 38:49–95, March 1996.

[WX08] A. Wigderson and D. Xiao. Derandomizing the Ahlswede-Winter matrix-valued Chernoff bound using pessimistic estimators, and applications. *Theory of Computing*, 4(3):53–76, 2008.

[WBA+07] E. L. Wong, P. Balasubramanian, L. Alvisi, M. G. Gouda, and V. Shmatikov. Truth In Advertising: Lightweight Verification of Route Integrity. In *PODC*, 2007.

[Xia09] D. Xiao. On basing **ZK** $\neq$ **BPP** on the hardness of PAC learning. In *In Proc. CCC '09*, pages 304–315, 2009.

[Yao82] A. C. Yao. Theory and Applications of Trapdoor Functions. In *Proc. 23rd FOCS*, pages 80–91. IEEE, 1982.

[YN77] D. B. Yudin and A. S. Nemirovski. Informational complexity and efficient methos for solving complex extremal problems. *Matekon*, 13:25–45, 1977.

Appendix A

Appendix

A.1 PSPACE and oracles

Proposition A.1.1. $\mathsf{QBF}_*^{\mathcal{R}}$ *is* $\mathbf{PSPACE}_*^{\mathcal{R}}$*-complete.*

Proof. $\mathsf{QBF}_*^{\mathcal{R}} \in \mathbf{PSPACE}_*^{\mathcal{R}}$: this immediate because the proof that $\mathsf{QBF} \in \mathbf{PSPACE}$ relativizes. On input φ, M_1 takes φ and outputs all the z such that φ contains a $\mathcal{R}_z$ gate to obtain $z_1, \ldots, z_m$. M_2 then simply decides φ using access to the $\mathcal{R}_{z_i}$ gates. This runs in space $O(n^2)$ and therefore time $2^{O(n^2)}$, see *e.g.* the analysis in [AB09].

All $L \in \mathbf{PSPACE}_*^{\mathcal{R}}$ reduce to $\mathsf{QBF}_*^{\mathcal{R}}$: recall the proof that QBF is complete for $\mathbf{PSPACE}$ (see *e.g.* [AB09]). For a $\mathbf{PSPACE}$ machine M with space bound $p(n)$ and an input x, we look at the configuration graph of M on input x. A state of the configuration graph is describable by a string of size $O(p(n))$. Furthermore, there is a $O(p(n))$ size formula $\phi_{M,x}$ that describes edges in the configuration graph: namely, given $S, S' \in \{0,1\}^{p(n)}$, $\phi_{M,x}(S, S') = 1$ iff S' follows from one step of the computation of M starting with configuration S. The QBF formula is constructed recursively by

contracting paths in the configuration graph: we initialize $\psi_1 = \phi$ and define

$$\psi_i(S, S') = \exists S'', \forall T_1, T_2, (T_1 = S \wedge T_2 = S'') \vee (T_1 = S'' \wedge T_2 = S') \Rightarrow \psi_{i-1}(T_1, T_2)$$

and the final output formula is $\psi_{p(n)}(S_0, S_a)$ where S_0 is the initial configuration and S_a is an accepting final configuration. One can check that $|\psi_{p(n)}(S_0, S_a)| = O(p(n)^2)$.

To generalize this reduction to $\mathbf{PSPACE}^{\mathcal{R}}_*$, on input x our reduction first uses M_1 to obtain $z_1, \ldots, z_m$. Now, it produces the formula $\phi_{M,x}$, which contains only (say) NAND gates and gates of the form $\mathcal{R}_{z_i}$. Then, run the same reduction as in the **PSPACE** case, which gives us the final formula $\psi_{p(n)}(S_0, S_a)$ which contains only $\mathcal{R}_z$ gates with explicit z (*i.e.* those obtained from M_1).

■

We use the fact that for any $\mathbf{PSPACE}^{\mathcal{O}}$ relation R, a $\mathbf{PSPACE}^{\mathcal{O}}$ oracle can count the number of satisfying pairs $\{(x, y) \mid R(x, y) = 1\}$ simply by enumerating over all pairs and checking the relation. We use this to show the following two facts. First, $\mathbf{PSPACE}^{\mathcal{O}}$ is able to invert itself:

Proposition A.1.2. *There is an efficient oracle algorithm A that, for every $\mathcal{O}$, $A^{\mathbf{PSPACE}^{\mathcal{O}}}$ takes input a circuit $C : \{0,1\}^\ell \to \{0,1\}^m$ with oracle gates and a string $y \in \{0,1\}^m$, and outputs a uniform element of the set $\{x \mid C^{\mathbf{PSPACE}^{\mathcal{O}}}(x) = y\}$ with probability at least $1 - 2^{-|y|}$, and outputs a special failure symbol $\perp$ with the remaining probability.*

Proof. The computation of C on inputs of length ℓ can be expressed as a polynomial-size $\mathsf{QBF}^{\mathcal{O}}$, and so we can use a $\mathbf{PSPACE}^{\mathcal{O}}$ oracle to compute $s = |(C^{\mathbf{PSPACE}^{\mathcal{O}}})^{-1}(y)|$. Now pick a random number $i \leftarrow_{\mathrm{R}} [s]$ and use the $\mathbf{PSPACE}^{\mathcal{O}}$ oracle to output the i'th lexicographically ordered string in $f^{-1}(y)$. There is some probability of failure

because sampling a number in $[s]$ may have a probability of failure if s is not a power of 2, but this can be made to be smaller than $2^{-|y|}$ by repeating the procedure. ■

Second, $\mathbf{PSPACE}^{\mathcal{O}}$ is able to find "heavy" outputs of a $\mathbf{PSPACE}^{\mathcal{O}}$ computation, say over the uniform distribution of inputs (in the following, think of D as being the equality predicate; we will state it more generally because of how we use it in our proofs):

Proposition A.1.3. *There is an efficient oracle algorithm A that, for every $\mathcal{O}$, $A^{\mathbf{PSPACE}^{\mathcal{O}}}$ takes input two oracle circuits $C : \{0,1\}^{\ell} \to \{0,1\}^m$ and circuit $D : \{0,1\}^m \times \{0,1\}^n \to \{0,1\}$ computing a predicate, and a unary string 1^p and outputs a set*

$$S = \left\{ y \;\middle|\; \Pr_{x \leftarrow_R U_\ell} \left[D^{\mathbf{PSPACE}^{\mathcal{O}}}(C^{\mathbf{PSPACE}^{\mathcal{O}}}(x), y) = 1 \right] \geq 1/p \right\}$$

Proof. Since $\mathbf{PSPACE}^{\mathcal{O}}$ is capable of counting $\mathbf{PSPACE}^{\mathcal{O}}$ relations, A simply iterates over all $x \in \{0,1\}^m, y \in \{0,1\}^n$ and outputs all y such that the number of x such that $D^{\mathbf{PSPACE}^{\mathcal{O}}}(C^{\mathbf{PSPACE}^{\mathcal{O}}}(x), y) = 1$ is larger than $2^n/p$. There can be at most p such y, so the procedure runs in polynomial space. ■

A.2 Chernoff bounds for smooth distributions

The standard Chernoff shows that the empirical average of many samples drawn from a distribution deviates from the mean of the distribution with exponentially small probability. We use the fact that this also holds for weighted empirical averages, as long as the weights are relatively smooth.

Lemma A.2.1 (Generalized Chernoff bound)**.** *Let D be a distribution over a finite universe U such that $\max_{u \in U} \Pr[D = u] \leq 1/k$ (equivalently, it has min-entropy*

$H_\infty(D) \geq \log k$). *Let F be a distribution on functions $f : U \to \{0,1\}$. Let $\mu = \mathbb{E}_{D,F}[F(D)]$ and let $\mu_u = \mathbb{E}_F[F(u)]$. Then*

$$\Pr_F \left[\mathbb{E}_D[F(D)] > \mu + \gamma\right] < e^{-\gamma^2 k/2}$$

Proof. We derive that for any positive constant t:

$$\begin{aligned}
\Pr_F \left[\mathbb{E}_D[F(D)] > \mu + \gamma\right] &= \Pr_F \left[e^{t(k\mathbb{E}_D[F(D)] - k\mu)} > e^{tk\gamma}\right] \\
&\leq e^{-tk\gamma}\mathbb{E}_F \left[e^{t(k\mathbb{E}_D[F(D)] - k\mu)}\right] \\
&\leq e^{-tk\gamma}\mathbb{E}_F \left[e^{t(k\mathbb{E}_D[F(D) - \mu_D])}\right] \\
&\leq e^{-tk\gamma}\mathbb{E}_F \left[e^{t(\sum_{u \in \mathrm{supp}(D)} F(u) - \mu_u)}\right] \qquad \textit{(using } \Pr[D = u] \leq 1/k\textit{)} \\
&= e^{-tk\gamma} \prod_{u \in \mathrm{supp}(D)} \mathbb{E}_F \left[e^{t(F(u) - \mu_u)}\right] \\
&\leq e^{-tk\gamma + t^2 k} \qquad \textit{(using } |\mathrm{supp}(D)| \geq k \textit{ plus Taylor expansion)} \\
&= e^{-\gamma^2 k/2}
\end{aligned}$$

where the last line follows from setting $t = \gamma/2$. ∎

A.3 Protocols for set sizes

In this section, we present constant-round protocols for lower- and upper-bounding set sizes.

Lemma A.3.1 (Lower-bound protocol [GS89])**.** *Let $S \subset \{0,1\}^n$ be a set whose membership can be verified in* poly(n) *time. Then for every $\delta, \varepsilon \geq 1/\mathrm{poly}(n)$, there exists an* **AM** *protocol where the efficient verifier V is given an integer s such that:*

- *If $|S| \geq s$, then there is an honest prover that makes V accept with probability at least $1 - \delta$.*

- *If $|S| \leq (1-\varepsilon)s$, then for any (possibly malicious and unbounded) prover, V accepts with probability at most δ.*

Lemma A.3.2 (Upper-bound protocol [For87, AH91])**.** *Let $S \subset \{0,1\}^n$ be a set whose membership can be verified in* poly(n) *time. Then for every $\delta, \varepsilon \geq 1/\text{poly}(n)$, there exists an* **AM** *protocol where the efficient verifier V is given an integer s and a random $x \leftarrow_R S$ unknown to the prover such that:*

- *If $|S| \leq s$, then there is an honest prover that makes V accept with probability at least $1-\delta$.*
- *If $|S| \geq (1+\varepsilon)s$, then for any (possibly malicious and unbounded) prover, V accepts with probability at most $1-\delta-\varepsilon$.*

Observe that the gap between between the honest and malicious case is small (only ε). This is not a problem for our applications since we will usually run $o(1/\delta)$ such lower-bound protocols in parallel, which means that if the prover is honest then none of the parallel executions rejects, while if the prover is malicious then he can cheat on at most a $O(1/\varepsilon)$ fraction of the executions without getting caught. This soundness suffices for our applications; see the application in the proof of the following Lemma A.4.1 or the papers [BT06, AGGM06] for more details.

A.4 $\mathsf{SD} \in \mathbf{AM} \cap \mathbf{coAM}$

The following lemma shows that $\mathsf{SD}_{\alpha,\beta} \in \mathbf{AM} \cap \mathbf{coAM}$ even when $\beta - \alpha = 1/\text{poly}(n)$, *i.e.* the gap between β, α is small. This can be derived from known results, for example from [GVW01], since $\mathsf{SD}_{\alpha,\beta}$ can be decided with constant completeness and soundness error by a 2-message protocol where the prover sends $O(\log n)$ bits (simply

repeat the **SZK** protocol for $\mathsf{SD}_{\alpha,\beta}$ for $O(\log n)$ times in parallel), which by definition means $\mathsf{SD}_{\alpha,\beta} \in \mathbf{AM}$, and then applying Theorem 3.7 of [GVW01] implies that $\mathsf{SD}_{\alpha,\beta} \in \mathbf{coAM}$. However, here we present a direct and self-contained protocol.

Lemma A.4.1. *For any polynomial $p(n)$ and for any α, β satisfying $\alpha - \beta \geq 1/p(n)$, it holds that $\mathsf{SD}_{\alpha,\beta} \in \mathbf{AM} \cap \mathbf{coAM}$.*

Proof. We will show that it is possible to approximate the statistical distance $\Delta(X_0, X_1)$ up to accuracy $\frac{\alpha-\beta}{2}$ using an **AM** protocol.

Lemma A.4.2. *There exists a protocol $\mathsf{SD}_{\mathsf{approx}}$ where P, V get as input a pair of polynomial-size circuits X_0, X_1 mapping $\{0,1\}^m \to \{0,1\}^n$, a number γ, and a confidence parameter $1/2 > \eta > 0$ such that V runs in time $\mathrm{poly}(n, 1/\eta)$, and it holds that:*

1. *If $\mathsf{SD}(X_0, X_1) = \gamma$, then the honest prover P convinces the verifier V to accept with probability $1 - \eta$.*
2. *If $|\mathsf{SD}(X_0, X_1) - \gamma| > \eta$, then for all (possibly malicious and unbounded) provers P^*, the verifier accepts with probability at most η.*

This lemma implies that $SD_{\alpha,\beta} \in \mathbf{AM} \cap \mathbf{coAM}$, since the verifier could first use the protocol to approximate the statistical distance and then decide (X_0, X_1) based on this approximation. ■

Proof of Lemma A.4.2. We begin with some notation. Let X_b be the distribution obtained by first sampling $b \leftarrow_{\mathrm{R}} \{0,1\}$ and then sampling $x \leftarrow_{\mathrm{R}} X_b$. Notice X_b can

be efficiently sampled using $m+1$ bits. Define

$$(X_0)^{-1}(x) = \{r \mid x = X_0(r)\}$$
$$(X_1)^{-1}(x) = \{r \mid x = X_1(r)\}$$
$$(X_b)^{-1}(x) = \{(b, r) \mid x = X_b(r)\}$$

Observe that for all $x \in \text{supp}(X_b)$, it holds that $|(X_b)^{-1}(x)| = |(X_0)^{-1}(x)| + |(X_1)^{-1}(x)|$.

The protocol satisfying the lemma is specified as follows. Let $t = \frac{8}{\eta^2}\log(2/\eta)$:

1. V samples $x_1, \ldots, x_t$ from the distribution X_b. V sends $x_1, \ldots, x_t$ to P.
2. P responds with integers $(s_{0,1}, \ldots, s_{0,t}), (s_{1,1}, \ldots, s_{1,t}), (r_1, \ldots, r_t)$.
3. V checks that for all $i \in [t]$, it holds that $r_i = s_{0,i} + s_{1,i}$, if not V rejects.
4. In parallel, P, V engages in the following upper/lower bound protocols (Lemma A.3.1 and Lemma A.3.2) for each $i \in [t]$ with parameters $\delta = \frac{\eta}{8tn}$ and $\varepsilon = \frac{\eta}{16}$:
   (a) A lower-bound protocol that $|(X_0)^{-1}(x_i)| \geq s_{0,i}$.
   (b) A lower-bound protocol that $|(X_1)^{-1}(x_i)| \geq s_{1,i}$.
   (c) A lower-bound protocol that $|(X_b)^{-1}(x_i)| \geq r_i$.
   (d) An upper-bound protocol that $|(X_b)^{-1}(x_i)| \leq r_i$.

   Observe that for the upper bound protocol the verifier V has a secret random sample in the preimage, namely the preimage it used to sample x_i.
5. If none of the upper/lower bound protocols reject, then V checks whether

$$\left| \frac{1}{t} \sum_{i=1}^{t} \frac{|s_{0,i} - s_{1,i}|}{r_i} - \gamma \right| \leq \frac{\eta}{2}$$

   If so V accepts, otherwise it rejects.

We observe first that if $s_{0,i} = |(X_0)^{-1}(x_i)|, s_{1,i} = |(X_1)^{-1}(x_i)|, r_i = |(X_b)^{-1}(x_i)|$, then it holds that

$$\begin{aligned}\frac{|s_{0,i} - s_{1,i}|}{r_i} &= \frac{\frac{1}{2}|s_{0,i} - s_{1,i}|}{\frac{1}{2}(s_{0,i} + s_{1,i})} \\ &= \frac{\frac{1}{2}|\Pr[X_0 = x_i] - \Pr[X_b = x_i]|}{\frac{1}{2}\Pr[X_0 = x_i] + \frac{1}{2}\Pr[X_1 = x_i]} \\ &= \frac{1}{2}\frac{|\Pr[X_0 = x_i] - \Pr[X_b = x_i]|}{\Pr[X_b = x_i]}\end{aligned}$$

Furthermore, we observe that

$$\begin{aligned}\Delta(X_0, X_1) &= \frac{1}{2}\sum_{x \in \mathrm{supp}(X_0) \cup \mathrm{supp}(X_1)} |\Pr[X_0 = x] - \Pr[X_1 = x]| \\ &= \sum_{x \in \mathrm{supp}(X_0) \cup \mathrm{supp}(X_1)} \frac{1}{2}\Pr[X_b = x]\frac{|\Pr[X_0 = x] - \Pr[X_1 = x]|}{\Pr[X_b = x]} \\ &= \mathbb{E}_{x \leftarrow_{\mathrm{R}} X_b}\left[\frac{1}{2}\frac{|\Pr[X_0 = x] - \Pr[X_1 = x]|}{\Pr[X_b = x]}\right]\end{aligned}$$

Therefore, it follows that:

$$\mathbb{E}_{x_i \leftarrow_{\mathrm{R}} X_b}\left[\frac{|s_{0,i} - s_{1,i}|}{r_i}\right] = \mathbb{E}_{x_i \leftarrow_{\mathrm{R}} X_b}\left[\frac{1}{2}\frac{|\Pr[X_0 = x_i] - \Pr[X_1 = x_i]|}{\Pr[X_b = x_i]}\right] = \Delta(X_0, X_1) \tag{D.1}$$

Completeness: in this case, the prover gives the honest sizes $s_{0,i} = |(X_0)^{-1}(x_i)|, s_{1,i} = |(X_1)^{-1}(x_i)|, r_i = |(X_b)^{-1}(x_i)|$. Because the sizes are honest, this means by the completeness conditions of Lemma A.3.1 and Lemma A.3.2 that the probability that any of these protocols rejects is at most $\delta t = \frac{\eta}{8n} < \eta/2$. Furthermore, since each sample x_i is drawn from X_b and from Equation D.1 we have $\mathbb{E}[\frac{|s_{0,i} - s_{1,i}|}{r_i}] = \Delta(X_0, X_1) = \gamma$, it follows by a Chernoff bound that

$$\Pr_{x_1, \ldots, x_t}\left[\left|\frac{1}{t}\sum_{i=1}^{t}\frac{|s_{0,i} - s_{1,i}|}{r_i} - \gamma\right| > \frac{\eta}{2}\right] \leq 2^{-\eta^2 t/8}$$

which, by our choice of $t = \frac{8}{\eta^2}\log(2/\eta)$, is bounded by $\eta/2$. Therefore, the probability that V rejects is at most η.

Soundness: by a Chernoff bound, we have that

$$\Pr_{x_1,\ldots,x_t}\left[\left|\frac{1}{t}\sum_{i=1}^{t}\frac{||(X_0)^{-1}(x_i)|-|(X_1)^{-1}(x_i)||}{|(X_b)^{-1}(x_i)|}-\Delta(X_0,X_1)\right|>\frac{\eta}{2}\right]\leq 2^{-\eta^2 t/8}$$

which is bounded by $\eta/2$, so condition on the event that this does not hold, *i.e.* the event that

$$\left|\frac{1}{t}\sum_{i=1}^{t}\frac{||(X_0)^{-1}(x_i)|-|(X_1)^{-1}(x_i)||}{|(X_b)^{-1}(x_i)|}-\Delta(X_0,X_1)\right|\leq\frac{\eta}{2} \tag{D.2}$$

Cheating lower bounds: if there exists i such that

$$|(X_0)^{-1}(x_i)|<(1-\varepsilon)s_{0,i},\qquad |(X_1)^{-1}(x_i)|<(1-\varepsilon)s_{1,i},\qquad |(X_b)^{-1}(x_i)|<(1-\varepsilon)r_i$$

then one of the lower bound protocols will reject with probability $1-\delta>1-\eta/2$. So suppose not, and that for all $i\in[t]$ that

$$|(X_0)^{-1}(x_i)|\geq(1-\varepsilon)s_{0,i},\quad |(X_1)^{-1}(x_i)|\geq(1-\varepsilon)s_{1,i},\quad |(X_b)^{-1}(x_i)|\geq(1-\varepsilon)r_i \tag{D.3}$$

Cheating upper bounds: now consider if the prover tries to cheat in some of the upper bounds. Let S denote the set of i such that $|(X_b)^{-1}(x_i)|>(1+\varepsilon)r_i$. We claim that if $|S|>\frac{8}{\varepsilon}\log\frac{2}{\eta}$, then the verifier accepts all these cheating executions is $\leq\eta/2$. The probability that one execution of the upper bound protocol accepts is at most $(1-\varepsilon-\delta)$, and since the parallel executions are independent and because $\delta\ll\varepsilon$, it follows that the probability that they all accept is at most

$$(1-\varepsilon-\delta)^{|S|}\leq(1-\varepsilon-\delta)^{\frac{8}{\varepsilon}\log\frac{2}{\eta}}\leq 2^{\log\frac{2}{\eta}}\leq\eta/2$$

Therefore, we may assume that

$$|S|\leq\tfrac{8}{\varepsilon}\log\tfrac{2}{\eta} \tag{D.4}$$

Notice that this means that $|S|/t\leq\varepsilon/2$.

Finally in order for V to accept it must be the case that

$$\left|\frac{1}{t}\sum_{i=1}^{t}\frac{|s_{0,i}-s_{1,i}|}{r_i}-\gamma\right|\leq\frac{\eta}{2} \tag{D.5}$$

But since we assumed that Inequality D.2 holds, namely that the honest preimage sizes give a good estimation of the statistical distance, and because γ is far from the true statistical distance, in order for P to make V accpt it must be that P's claims must be far from the honest preimage sizes. That is, applying the triangle inequality to Inequality D.2 and Inequality D.5, P must give $s_{0,i}, s_{1,i}, r_i$ satisfying

$$\left|\frac{1}{t}\sum_{i=1}^{t}\frac{||(X_0)^{-1}(x_i)|-|(X_1)^{-1}(x_i)||}{|(X_b)^{-1}(x_i)|}-\frac{1}{t}\sum_{i=1}^{t}\frac{|s_{0,i}-s_{1,i}|}{r_i}\right|>\frac{\eta}{2} \tag{D.6}$$

We will show that this contradicts Inequality D.3 and Inequality D.4. First, because $|(X_0)^{-1}(x_i)|+|(X_1)^{-1}(x_i)|=|(X_b)^{-1}(x_i)|$ and $s_{0,i}+s_{1,i}=r_i$, we have that for all $i\in[t]\setminus S$

$$|(X_0)^{-1}(x_i)|=|(X_b)^{-1}(x_i)|-|(X_1)^{-1}(x_i)|\leq(1+\varepsilon)r_i-(1-\varepsilon)s_{1,i}=s_{0,i}+\varepsilon(r_i+s_{1,i}) \tag{D.7}$$

$$|(X_1)^{-1}(x_i)|=|(X_b)^{-1}(x_i)|-|(X_0)^{-1}(x_i)|\leq(1+\varepsilon)r_i-(1-\varepsilon)s_{0,i}=s_{1,i}+\varepsilon(r_i+s_{0,i}) \tag{D.8}$$

For each $i\in[t]\setminus S$, it holds that

$$\begin{aligned}
\frac{|(X_0)^{-1}(x_i)|-|(X_1)^{-1}(x_i)|}{|(X_b)^{-1}(x_i)|} &\leq \frac{s_{0,i}+\varepsilon(r_i+s_{1,i})-(1-\varepsilon)s_{1,i}}{(1-\varepsilon)r_i}\\
&= \frac{s_{0,i}-s_{1,i}+\varepsilon(r_i+2s_{1,i})}{(1-\varepsilon)r_i}\\
&\leq \frac{|s_{0,i}-s_{1,i}|+3\varepsilon r_i}{(1-\varepsilon)r_i} \qquad (Since\ s_{1,i}\leq r_i)\\
&< (1+2\varepsilon)\frac{|s_{0,i}-s_{1,i}|}{r_i}+3\tfrac{\varepsilon}{1-\varepsilon}\\
&< \frac{|s_{0,i}-s_{1,i}|}{r_i}+6\varepsilon \qquad (Since\ |s_{0,i}-s_{1,i}|\leq r_i)
\end{aligned}$$

Here we have used the fact that $\varepsilon = \eta/16 < 1/32$ (since $\eta < 1/2$), which implies that $\frac{1}{1-\varepsilon} \leq 1 + 2\varepsilon$. We can repeat the above derivation with $0, 1$ interchanged, so we may deduce that for all $i \in [t] \setminus S$ it holds that

$$\left| \frac{||(X_0)^{-1}(x_i)| - |(X_1)^{-1}(x_i)||}{|(X_b)^{-1}(x_i)|} - \frac{|s_{0,i} - s_{1,i}|}{r_i} \right| \leq 6\varepsilon$$

Finally, we can derive that

$$\begin{aligned}
&\left| \frac{1}{t} \sum_{i=1}^{t} \frac{||(X_0)^{-1}(x_i)| - |(X_1)^{-1}(x_i)||}{|(X_b)^{-1}(x_i)|} - \frac{1}{t} \sum_{i=1}^{t} \frac{|s_{0,i} - s_{1,i}|}{r_i} \right| \\
&\quad \leq \left| \frac{1}{t} \sum_{i \in S} \frac{||(X_0)^{-1}(x_i)| - |(X_1)^{-1}(x_i)||}{|(X_b)^{-1}(x_i)|} - \frac{1}{t} \sum_{i \in S}^{t} \frac{|s_{0,i} - s_{1,i}|}{r_i} \right| \\
&\qquad + \left| \frac{1}{t} \sum_{i \in [t] \setminus S} \frac{||(X_0)^{-1}(x_i)| - |(X_1)^{-1}(x_i)||}{|(X_b)^{-1}(x_i)|} - \frac{1}{t} \sum_{i \in [t] \setminus S}^{t} \frac{|s_{0,i} - s_{1,i}|}{r_i} \right| \\
&\quad \leq \frac{|S|}{t} + \frac{1}{t} \sum_{i \in [t] \setminus S} 6\varepsilon \\
&\quad \leq 6.5\varepsilon \\
&\quad < \eta/2
\end{aligned}$$

which contradicts Inequality D.6. Therefore the protocol is sound, since either he cheats too much in the upper or lower bound protocols in which case he is caught with probability $1 - \eta$, or the verifier rejects in step 5 of the protocol. ■