論文 / 著書情報
Article / Book Information

題目(和文)	
Title(English)	Learning Theory in Heuristica and Pessiland
著者(和文)	七島幹人
Author(English)	Mikito Nanashima
出典(和文)	学位:博士(理学), 学位授与機関:東京工業大学, 報告番号:甲第12330号, 授与年月日:2023年3月26日, 学位の種別:課程博士, 審査員:伊東 利哉,渡辺 治,田中 圭介,鹿島 亮,安永 憲司
Citation(English)	Degree:Doctor (Science), Conferring organization: Tokyo Institute of Technology, Report number:甲第12330号, Conferred date:2023/3/26, Degree Type:Course doctor, Examiner:,,,,
学位種別(和文)	博士論文
Type(English)	Doctoral Thesis

Learning Theory in Heuristica and Pessiland

Mikito Nanashima

Department of Mathematical and Computing Science

School of Computing

A Dissertation

Submitted for the Degree of Doctor of Philosophy

at Tokyo Institute of Technology

Under the Supervision of Professor Toshiya Itoh

Abstract

Heuristica and Pessiland, introduced by Impagliazzo (CCC'95), are possibilities of our world that are consistent with but far from our current knowledge. Heuristica is a world in which NP-problems are hard in the worst case but easy in the average case. Pessiland is a world in which NP-problems are hard in the average case, and there is no secure cryptography, particularly no central cryptographic primitive called one-way functions. Excluding the possibilities of Heuristica and Pessiland, i.e., basing a one-way function on the worst-case hardness of NP, is one of the most important challenges for unifying the central notions in computational complexity theory and cryptography. One natural approach for excluding Heuristic and Pessiland is to investigate algorithmic aspects of these possible worlds; e.g., to exclude Pessiland, we need to magnify the algorithmic aspects of Pessiland to efficient heuristic schemes that solve every NP-problem on average.

In this thesis, we study algorithmic aspects of Heuristica and Pessiland from the perspective of learning. This is motivated by the known facts that both the worst-case hardness of NP and the existence of one-way functions are characterized by the hardness of variants of Probability Approximately Correct (PAC) learning, as proved by Pitt and Valiant (J. ACM, 1990) and Blum, Furst, Kearns, and Lipton (Crypto'93), respectively. Our results are mainly classified into the following three topics.

I. Learning in Heuristica. We investigate connections between learning and average-case complexity of NP. The original PAC learning model, introduced by Valiant (J. Commun. ACM, 1984), has a worst-case nature due to the requirement that a learner must learn *all* functions in the target class under *all* unknown example distributions, and the feasibility thus seems not to follow from the average-case easiness of NP. We falsify this intuition by showing that the feasibility of worst-case PAC learning (even a more challenging task called agnostic learning) is indeed derived from the errorless average-case easiness of NP when the unknown example distributions are efficiently sampled by circuits. Namely, we obtain a reduction from worst-case learning to average-case NP under the additional computational assumption on example distributions.

II. Learning in Pessiland. We establish a robust and unified theory of average-case learning, which shows the equivalence between the non-existence of one-way functions and the feasibility of various average-case learning tasks, including average-case agnostic learning, average-case distributional learning, weak average-case learning with membership queries under the uniform example distribution, and learning adaptively changing distributions without knowledge of the distributions. It is worthy of note that the last learning task was previously thought to be *information-theoretically impossible* (Naor and Rothblum, ICML'06). We thus obtain a one-way function whose security is based on the intractability of a seemingly impossible average-case learning. In addition, we also present other duality results between learning and cryptography: (i) characterization of the existence of one-way functions by the average-case hardness of an NP-complete learning problem called MINLT in a well-studied average-case setting and (ii) new learning-theoretic characterizations of im-

portant cryptographic primitives, such as auxiliary-input one-way functions and polynomial-stretch pseudorandom generators computable in constant parallel time.

III. New and Improved Oracle Separations. We study the limitations of the standard proof framework called relativizing proofs. Here, relativizing proofs are proofs that hold even with additional query access to an arbitrary oracle, and almost all results in theoretical computer science shown by reductions, including the main results in this thesis, can be shown by relativizing proofs. We show that, for further improvements of our results, we require profoundly new *non-relativizing* proofs. First, we show a strong oracle separation between worst-case and average-case complexity. Our result drastically improves the previous separation result by Impagliazzo (CCC'11) and is tight because it matches the known upper bound shown by Hirahara (STOC'21). The oracle separation shows strong evidence that the additional computational assumption on example distributions is inevitable for any relativizing reduction from worst-case learning to average-case NP. Second, we show a new oracle separation between errorless and error-prone average-case complexity, which was asked in the seminal paper by Impagliazzo (CCC'95). We thus resolve the problem that had been open for more than two decades. The second separation result shows strong evidence that (i) the errorless condition is inevitable for any relativizing reduction from worst-case learning to average-case NP, and (ii) average-case requirements cannot be improved to worst-case even partially in our unified theory of average-case learning unless we develop a non-relativized technique.

Besides the topics above, we obtain a new characterization of the hardness of PAC learning by an auxiliary-input cryptographic primitive, more precisely, an auxiliary-input hitting set generator with a local condition. We further show that, for basing one-way functions on NP-hardness, it suffices to base such auxiliary-input cryptographic primitives on NP-hardness by a restricted form of *nonadaptive black-box* security reductions. Optimistically, this suggests a new approach towards excluding Heuristica and Pessiland, which reduces constructing a one-way function to constructing an auxiliary-input cryptographic primitive whose security condition is much more relaxed and even weaker than the hardness of PAC learning.

Acknowledgements

I express my sincere gratitude to my academic adviser Toshiya Itoh for his kind and enduring support. Since I knew almost nothing about research activities, he has always been there and provided me with advice and guidance. His support is the most crucial factor in completing my degree program successfully and has helped me acquire my foundational academic skills.

I sincerely thank Osamu Watanabe, who was also my adviser during bachelor's and master's courses. He sparked my interest in the theory of computing and encouraged me to pursue a Ph.D. He sometimes gave me strong encouragement, and I think it will continue to support me throughout my life as a researcher.

I want to thank Shuichi Hirahara from the bottom of my heart for all the helpful and exciting discussions. He was like a foster mentor for me, and many parts of this thesis were made as joint work with him. I think collaborating with him is the most fortunate part of my Ph.D. program. Without this invaluable opportunity, my understanding of the research subject was shallow and narrow, and this thesis would be much more miserable.

I am grateful for the financial support from JST ACT-X (JPMJAX190M) and the JSPS research fellowship for young scientists. I also want to thank ACT-X for giving me the opportunity to communicate with many distinguished young researchers. I would like to sincerely thank the research supervisor Kenichi Kawarabayashi, the area advisers, especially Tsuyoshi Takagi and Tomoyuki Morimae, and the manager Hitoshi Yashio. I also want to sincerely thank Ryuhei Mori for encouraging me to apply for ACT-X.

I have had many great opportunities to interact with many excellent researchers during my degree program. The discussions with them have helped me deepen and broaden my knowledge. I particularly want to thank Rahul Ilango, Zhenjian Lu, and Igor Carboni Oliveira for the very exciting collaboration. I also want to thank Kaito Fujii, Ryokichi Tanaka, Nobutaka Shimizu, Ryoshun Oba, and Shu Kanazawa for fruitful regular discussions. I also want to express my appreciation to all researchers who have contributed to my research in any form, including discussion at the conferences and feedback in the reviewing processes.

I sincerely thank Yuko Hirauchi for her help with various official procedures. I am very grateful that her support allowed me to concentrate on my research. I also want to thank all of the past and current members of the laboratory, especially Steven Ge. He is a very nice person, and without him, I was literally alone in the laboratory.

Lastly, I express my deepest gratitude to my parents, Nanashima Masato and Megumi, and my grandparents, Go Toshiko and Tomohira, for all their encouragement and support since I was born.

Contents

Chapter 1

Introduction

NP-problems are one of the most influential discoveries in theoretical computer science. Roughly speaking, NP denotes the class of decision problems (i.e., a problem to which answered yes or no) that have efficiently verifiable witness when the answer is yes; e.g., a problem of determining whether a given linear system is satisfiable, and in this case, the witness for answering yes is the satisfying assignment itself. From the perspective of algorithm theory, solving NP efficiently (i.e., $\mathsf{P} = \mathsf{NP}$) is an ultimate goal because almost all problems we naturally care about are contained in NP. This is supported by, for instance, the fact that there are more than 500,000 studies[1] that discuss some NP-complete problems involved in various fields such as physics, biology, chemistry, and social theory. Here, an NP-complete problem is the most difficult problem in NP in the sense that if an NP-complete problem is solvable efficiently, then all NP-problems are also solvable efficiently. Imagine that, in the world of $\mathsf{P} = \mathsf{NP}$, we can solve all of such various problems efficiently, which must be a dream world of algorithms. By contrast, from the perspective of cryptography, the class NP has different importance as a source of computational hardness for preventing attacks from adversaries. It is well-known that we cannot hope for any secure cryptographic protocol without the computational hardness of NP because the existence of a secure cryptographic protocol (such as a public-key encryption scheme) implies the hardness of NP-problems (i.e., $\mathsf{P} \neq \mathsf{NP}$).

The argument above shows that we cannot simultaneously obtain secure cryptography and a powerful algorithm for an NP-complete problem. Then, can we guarantee *either* of them? More generally, can we establish a duality of a strong algorithmic consequence and secure cryptography? Achieving this is a desirable goal in the sense that it yields a win-win argument, i.e., our world is an algorithmically or cryptographically dream world.

Question 1.0.1. *Can we base the security of cryptography on the hardness of* NP*?*

Unfortunately, there is a huge gap between secure cryptography and the feasibility of NP in our current knowledge, and Question 1.0.1 is one of the central and longstanding open questions in theoretical computer science. Namely, at present, we cannot deny the tragic possibility that there is neither secure cryptography nor an efficient algorithm for solving NP-problems. In this thesis, we further discuss the gap inherent in Question 1.0.1 from two different perspectives.

[1] The result of searching for the term "NP complete" with Google Scholar in November 2022.

1.1 Average-Case Complexity and Impagliazzo's Five Worlds

Impagliazzo [Imp95] explained the current gaps between secure cryptography and the feasibility of NP more precisely based on the notion of *average-case complexity*, which was introduced by Levin [Lev86] in context of complexity theory and by Blum and Micali [BM84] and Yao [Yao82] in context of cryptography. Particularly, Impagliazzo [Imp95] presented five possibilities of our world that are consistent with our current knowledge, which often called *Impagliazzo's five worlds*[2]. Below, we highlight the five possible worlds. Note that we consider randomized Turing machines as a standard computational model, but we often do no care about the difference of computational models (e.g., deterministic Turing machine and circuits) when we discuss the notion of Impagliazzo's five worlds.

I. Algorithmica. This is a possible world in which every NP-problem is solvable efficiently. More precisely, Algorithmica is defined as a world in which $\mathsf{NP} \subseteq \mathsf{BPP}$ holds, where BPP is the class of decision problems solvable efficiently by randomized Turing machines.

II. Heuristica. This is a possible world in which an NP-problem is *not* solvable efficiently in the *worst-case* sense that every efficient algorithm fails to solve the problem for *some* instance, but every NP-problem is solvable in the *average-case* sense that an efficient algorithm solves the problem for *almost all* instances. More precisely, Heuristica is defined as a world in which $\mathsf{NP} \nsubseteq \mathsf{BPP}$ and $\mathsf{DistNP} \subseteq \mathsf{AvgBPP}$ hold, where DistNP and AvgBPP are the average-case analogues of NP and BPP, respectively. We also often consider another formulation of the class HeurBPP of average-case easy problems instead of AvgBPP. Roughly speaking, the difference between HeurBPP and AvgBPP lies in the requirement when an algorithm fails to solve problems (remember that, in the average-case setting, an algorithm is allowed to fail on a small fraction of instances). In the formulation of HeurBPP, we force an algorithm to solve a problem for almost all instances, i.e., we just discuss the success probability of the heuristic algorithm over instances. By contrast, in the formulation of AvgBPP, we consider an additional *errorless* requirement that an algorithm must output a special symbol $\perp$ for hard-to-solve instances and never output a wrong answer. Namely, $\mathsf{AvgBPP} \subseteq \mathsf{HeurBPP}$ holds. For more formal descriptions of DistNP, AvgBPP, and HeurBPP, refer to Section 2.2.

III. Pessiland. This is the worst one of all possible worlds, where we can hope for neither an efficient algorithm that solves an NP-problem on average nor secure cryptography. More precisely, Pessiland is defined as a world in which $\mathsf{DistNP} \nsubseteq \mathsf{AvgBPP}$ (or $\mathsf{DistNP} \nsubseteq \mathsf{HeurBPP}$) and there exists no *one-way function*, which is at the core of secure cryptography, as explained below.

IV. Minicrypt. This is the world in which we can construct minimum cryptographic protocols but cannot hope for public-key encryption schemes. More precisely, Minicrypt is defined as a world in which there is a one-way function, but there is no public-key encryption scheme. Again, a one-way function is a cryptographic primitive like an atom of secure cryptography in the sense that (i) it is necessary for almost all cryptographic schemes [cf. IL89; OW93] and (ii) it is sufficient for many cryptographic primitives such as a pseudorandom generator [HILL99; VZ12; HRV13], a pseudorandom function [GGM86], a digital signature [Rom90], a zero-knowledge proof [GMW91; NOV06] and a private-key encryption scheme [GGM84]. Namely, a theory of cryptography has been developed under the axiom that a one-way function exists, and Minicrypt is the world in which the axiom of cryptography is unconditionally verified.

[2]Recently, the sixth possible world "obfustopia," the world in which indistinguishably obfuscation is available, is often discussed, but this is out of the scope of this thesis.

V. Cryptomania. This is a world in which a public-key encryption scheme is available. Cryptomania seems widely and unconsciously believed to be our world even by those who are not familiar with computer science because we currently use many candidates for secure public-key encryption schemes while believing and wishing that the candidates are indeed secure.

Again, either of the possible worlds above exactly corresponds to our world, and Question 1.0.1 can be generalized as follows.

Question 1.1.1. *Identify which of Impagliazzo's five worlds corresponds to our world.*

Particularly, Question 1.0.1 corresponds to excluding Heuristica and Pessiland (and Minicrypt if we consider a public-key encryption scheme as the goal), which is the main concern of this thesis.

Question 1.1.2. *Can we exclude Heuristica and Pessiland?*

By definitions, we can observe that

$$\begin{aligned}\exists\text{public-key encryption} &\Rightarrow \exists\text{a one-way function} \\ &\Rightarrow \mathsf{DistNP} \nsubseteq \mathsf{HeurBPP} \Rightarrow \mathsf{DistNP} \nsubseteq \mathsf{AvgBPP} \Rightarrow \mathsf{NP} \nsubseteq \mathsf{BPP}. \quad (1.1)\end{aligned}$$

However, showing any converse is currently open and yields excluding the corresponding possible world. A natural approach towards excluding the false possible worlds is to investigate the algorithmic implications of the target world we try to exclude. For instance, to exclude Pessiland, we need to strengthen the algorithmic aspect (i.e., the non-existence of one-way functions) of Pessiland to efficient heuristic algorithms that solve every NP-problem on average.

1.2 Computational Learning Theory

Computational learning theory yields another perspective for discussing the gaps inherent in Question 1.0.1, which is a less common approach compared with the one based on the average-case complexity and Impagliazzo's five worlds in this context but still one of the central topics in theoretical computer science.

Computational learning theory is the field whose main concern lies in computational resources required for machine learning, which was opened by Valiant [Val84]. In the pioneering work, Valiant [Val84] introduced the Probably Approximately Correct (PAC) learning model as a natural formulation of learning some concepts from experience, and the learnability in the PAC learning model has been intensively studied over decades in the community. We refer the reader to the textbooks [KV94b; Wig19, Section 17] for further backgrounds of computational learning theory. Here, we briefly explain the PAC learning model.

PAC Learning Model [Val84]. A learner L is defined as a randomized algorithm (unless otherwise stated) and asked to learn a concept class $\mathscr{C}$ of Boolean-valued functions. More precisely, the learner L is asked to learn *all* functions $f \in \mathscr{C}$ under *all* example distributions $\mathcal{D}$ (where f and $\mathcal{D}$ are unknown for L) in the following sense: L is given an accuracy parameter $\epsilon \in (0,1)$ and sufficiently many samples $(x^1, f(x^1)), \ldots, (x^m, f(x^m))$ (we call each x^i an *example*, b^i a *label* of x^i, and f a *target function*), where the examples $x^1, \ldots, x^m$ are drawn identically and independently from the example distribution $\mathcal{D}$, and with high probability, needs to output a good hypothesis h in a hypothesis class $\mathscr{H}$ (the class of circuits unless otherwise stated) that approximates f within the accuracy error ϵ under the same example distribution, i.e., $\Pr_x[h(x) \neq f(x)] \leq \epsilon$, where x is drawn

from $\mathcal{D}$. We say that $\mathscr{C}$ is PAC learnable in polynomial time if there exists a polynomial-time learner L that learns $\mathscr{C}$ in the PAC learning model. Note that we may drop the description "in polynomial time" when the intention is clear in context. We also define a sample complexity of L as the number of samples L requires in worst case with respect to target functions in $\mathscr{C}$ and example distributions. We may allow L to have additional query access called *membership queries* to the unknown target function f as additional resources, where the learner can ask the value of $f(x)$ for an arbitrary example x.

PAC learning is deeply related to cryptography and NP-problems. In the pioneering paper, Valiant [Val84] has already observed that the existence of pseudorandom functions (whose existence is equivalent to the existence of one-way functions [GGM86; HILL99]) implies the hardness of PAC learning the class of polynomial-size circuits (denoted by $\mathsf{P/poly}$). Namely, PAC learning is generally hard in Minicrypt and Cryptomania. By contrast, Blumer, Ehrenfeucht, Haussler, and Warmuth [BEHW87] proved that $\mathsf{P/poly}$ is PAC learnable in polynomial time if $\mathsf{NP} \subseteq \mathsf{BPP}$. In fact, it is known that every concept class PAC learnable in polynomial time must be contained in $\mathsf{P/poly}$ [Sch90]. Therefore, in Algorithmica, every possibly learnable class is indeed PAC learnable in polynomial time. To sum up, we have that

$$\exists \text{a one-way function} \Rightarrow \mathsf{P/poly} \text{ is not PAC learnable in polynomial time } \Rightarrow \mathsf{NP} \nsubseteq \mathsf{BPP}. \tag{1.2}$$

However, proving any converse is a longstanding open question. Therefore, the hardness of PAC learning is also regarded as an intermediate notion between cryptography and the hardness of NP, as similar to the average-case hardness of NP.

Investigating the opposite directions of the implications of (1.2) provides one approach to resolve Questions 1.0.1 and 1.1.2 from the perspective of computational learning theory. Here, we discuss the usefulness of this approach by remarking that changing the requirements of PAC learning yields *characterizations* of the existence of one-way functions and the hardness of NP.

NP-hardness and Proper Learning. The PAC learners constructed under $\mathsf{NP} \subseteq \mathsf{BPP}$ through the framework by Blumer, Ehrenfeucht, Haussler, and Warmuth [BEHW87] indeed has an additional property: the learner for a concept class $\mathscr{C}$ always outputs a hypothesis in the same class $\mathscr{C}$ (i.e., $\mathscr{H} = \mathscr{C}$), which is called *proper* learning. In fact, the hardness of proper learning is known to often characterize the worst-case hardness of NP; e.g., Pitt and Valiant [PV88] proved that the class of 2-term DNF formulas (which is much simpler than $\mathsf{P/poly}$!) is *properly* PAC learnable in polynomial time if and only if $\mathsf{NP} \subseteq \mathsf{BPP}$.

Cryptography and Average-Case Learning. The original formulation of PAC learning has the worst-case nature, i.e., a learner needs to learn *all* target functions under *all* example distributions. Impagliazzo and Levin [IL90] claimed that an average-case formulation of learning characterizes the existence of one-way functions in very abstract arguments even without using the term "learning." In a follow-up study, Blum, Furst, Kearns, and Lipton [BFKL93] interpreted the ideas of [IL90] in context of PAC learning. Particularly, they introduced a natural average-case variant of PAC learning, we call it *the BFKL model* in this thesis[3], and proved the equivalence between the existence of one-way functions and the *average-case* hardness of PAC learning in the BFKL model. The BFKL model has two differences from the PAC learning model: (i) the target function f is selected according to some efficiently samplable distribution $\mathcal{F}$ fixed in advance, and

[3]The BFKL model is often called merely average-case learning or PAC learning on average. The reason why we use the unfamiliar term "BFKL model" is that we will consider other average-case formulations of learning and need to distinguish them from the formulation in [BFKL93] explicitly.

(ii) the example distribution $\mathcal{D}$ is also fixed to be some efficiently samplable distribution in advance. A learner in the BFKL model is constructed depending on each $\mathcal{F}$ and $\mathcal{D}$ and outputs a good hypothesis with high probability over the choice of the target function drawn from $\mathcal{F}$ and examples drawn from $\mathcal{D}$. They proved that there exists a one-way function if and only if an efficiently computable class $\mathscr{C}$ (e.g., $\mathsf{P/poly}$) is not PAC learnable on average in the BFKL model for some efficiently samplable $\mathcal{F}$ and $\mathcal{D}$. Notice that the worst-case requirements in the PAC learning model with respect to target functions and example distributions totally vanish in the BFKL model.

The characterizations above provide some learning-theoretic interpretations of the inherent gaps between the hardness of NP and cryptography.

Worst-case versus average-case. The original PAC learning has the two worst-case requirements with respect to target functions and example distributions, while the BFKL model has no worst-case requirement. Naturally, we can consider various intermediate ways to relax the worst-case requirements to average-case ones. For instance, what happens if we relax only one of the two worst-case requirements? What happens if we consider the worst-case requirement with respect to efficiently samplable distributions? By these step-by-step relaxations of worst-case requirements, the hardness of learning is conjectured to go throughout Heuristica and Pessiland.

Proper learning versus improper learning. The worst-case hardness of NP is characterized by the hardness of proper learning. However, extending the hypothesis class is known to change the hardness of learning drastically, where learning with a hypothesis class larger than a concept class is called *improper* learning. For instance, as already mentioned, Pitt and Valiant [PV88] proved that proper learning 2-term DNF formulas implies $\mathsf{NP} \subseteq \mathsf{BPP}$, but they also showed that *improper* learning the same class in feasible *unconditionally*. In cryptographic nature, it seems more natural to consider improper learning (particularly a hypothesis-free case where $\mathscr{H} = \mathsf{P/poly}$) because a malicious adversary does not need to consider a restricted class to learn secret information at all. In fact, to the best of our knowledge, all previous studies on cryptographic hardness of learning indeed discuss *improper* learning [e.g., Val84; Kha93; KV94a; AK95; AR16; OS17; DV21].

We remark that the average-case learner in [BFKL93] constructed from the non-existence of one-way functions is implicitly a *proper* learner. Nevertheless, their result and the NP-hardness of proper learning [PV88] do not imply excluding Pessiland. The reason of this is that we cannot reduce the average-case variant of NP-complete problems to proper learning in the BFKL model through the reduction [PV88], and we need to consider a more general average-case framework of learning that was not discussed before. This observation further motivates us to revisit the theory of average-case learning and investigate connections to Heuristica and Pessiland.

The arguments above are summarized in the following questions.

Question 1.2.1. *How is the nature of learning changed by gradually relaxing (i) the worst-case requirements to the average-case requirements and (ii) proper learning to improper learning?*

Question 1.2.2. *How strong are learning algorithms feasible in Heuristica and Pessiland?*

The purpose of this thesis is, through the questions above, to provide deep connections between average-case complexity, learning, and cryptography for new learning-theoretic insights into algorithmic aspects of Heuristica and Pessiland towards excluding them.

1.3 Overview of The Thesis

In this thesis, we will answer Questions 1.2.1 and 1.2.2 by providing new connections between learning, average-case complexity, and cryptography. It is worthy of note that many of our results are characterization results involved in learnability. Below, we overview the organization of this thesis with highlighting our contributions. We remark that some results are improved in this thesis compared with the corresponding ones in the original publications.

Note that we mainly use the term learning to refer to improper PAC learning in the hypothesis-free setting (i.e., the hypothesis class is the class of circuits) or its variant unless otherwise stated.

Chapter 2 Preliminaries

We introduce notations and basic knowledge used throughout the thesis, such as the basics of computational complexity theory, average-case complexity, learning theory, cryptography, and so on. Note that basic knowledge required only for one chapter (e.g., Fourier analysis) may be provided in each chapter as additional preliminaries. This chapter also provides a brief survey on learning-theoretic consequences in Algorithmica, where the main focus is NP-complete learning problems, such as proper learning [PV88, e.g.,] and MINLT [Hir22a]. The purpose of this survey is to provide knowledge helpful for the comparison between learning-theoretic aspects of Algorithmica and our results on learning-theoretic aspects of Heuristica and Pessiland.

Chapter 3 Learning in Heuristica

We investigate learning-theoretic aspects of Heuristica, i.e., learnability results derived from the average-case easiness of NP.

Section 3.1 Worst-Case Learning in Heuristica

PAC learning has the worst-case nature due to the requirement that a learner must learn *all* target functions under *all* example distribution, and thus average-case easiness of NP seems conceptually insufficient to derive PAC learnability. In this section, we falsify this intuition by showing that PAC learning P/poly is implied by the errorless average-case easiness of NP (i.e., DistNP $\subseteq$ AvgP) with an additional computational assumption that the unknown example distribution is efficiently samplable by circuits (i.e., P/poly-samplable). Our result can be regarded as the first reduction from worst-case learning to average-case NP, and it also holds in a more general learning model called *agnostic* learning. We also discuss that the necessity of the errorless condition and the additional computational assumption on example distributions. This section is based on the first half of the joint work with Shuichi Hirahara [HN21], which originally appeared in FOCS2021. We also remark that the assumption DistNP $\subseteq$ AvgP was further relaxed to DistNP $\subseteq$ AvgBPP in the subsequent work by Goldberg, Kabanets, Lu, and Oliveira [GKLO22] building upon our technique.

Section 3.2 Conditional Extrapolation

We introduce *conditional extrapolation* as another natural formulation of learning, which is inspired by the work by Impagliazzo and Levin [IL90]. Roughly speaking, conditional extrapolation is a task of, for a given conditional string x drawn from an efficiently samplable distribution $\mathcal{C}$, to sample a string subsequent to x under a target (efficiently samplable) distribution $\mathcal{D}$ (e.g., simulating how

Mozart continues the first eight measures written by Billie Eilish). We show that the feasibility of conditional extrapolation exactly characterizes the error-prone average-case easiness of NP, i.e., conditional extrapolation is feasible if and only if DistNP $\subseteq$ HeurBPP. This characterization result is a natural generalization of [IL89; IL90] and also introduced as a technical lemma in joint work with Shuichi Hirahara, Rahul Ilango, Zhenjian Lu, and Igor Carboni Oliveira [HILNO23].

Chapter 4 Learning in Pessiland I: A Unified Theory of Average-Case Learning

We identify learning-theoretic consequences in Pessiland by establishing a robust and unified theory of average-case learning, which shows the equivalence between the non-existence of one-way functions and the feasibility of various average-case learning tasks, including average-case agnostic learning, average-case distributional learning, weak average-case learning with membership queries under the uniform example distribution, and learning adaptively changing distributions without knowledge of the distributions. It is worthy of note that the last learning task was previously thought to be *information-theoretically impossible* by Naor and Rothblum [NR06]. We thus obtain a one-way function whose security is based on the intractability of a seemingly impossible average-case learning. Our framework is based on *universal extrapolation*, which was outlined without any formal description in the very first work by Impagliazzo and Levin [IL90] on learning-theoretic implications from the non-existence of one-way functions. Surprisingly, we show that the previously known learnability results obtained from the non-existence of one-way functions [BFKL93; NR06; NY19] can be improved, unified, and simplified by using the single algorithm of universal extrapolation. This chapter is based on an unpublished joint work with Shuichi Hirahara [HN23a].

Chapter 5 Learning in Pessiland II: More Dichotomies between Learning and Cryptography

We further investigate dichotomies between learning and cryptography, particularly in the BFKL model (i.e., a restricted model compared with Chapter 4).

Section 5.1 MINLT vs. One-Way Functions

MINLT, introduced by Ko [Ko91], is a learning problem that asks the minimum description size of the hypothesis consistent with a given sample set. Recently, Hirahara [Hir22a] proved the NP-hardness of MINLT by a non-relativizing proof. Therefore, investigating the feasibility of MINLT in average-case settings seems a hopeful approach to excluding Pessiland with breaking relativizing barriers. In this section, we prove that MINLT is feasible on average in the BFKL model under the non-existence of one-way functions with an additional standard derandomization assumption. Note that this result is not sufficient for excluding Pessiland because, to reduce the average-case variant of NP-complete problems to MINLT through Hirahara's reduction, we need to consider the more general framework of average-case learning discussed in Chapter 4. Our result is also regarded as a follow-up result of a recent line of studies [LP22; ACMTV21] in meta-complexity theory that characterizes the existence of one-way functions by the average-case hardness of NP-complete problems over uniformly random instances. Compared with these previous studies, we show the characterization of one-way functions in more general average-case settings (i.e., the BFKL model) than the uniform distribution, instead require an additional assumption of derandomization. The main result of this section originally appeared in the appendix of the manuscript [HN23a].

Section 5.2 Learning vs. Auxiliary-Input One-Way Functions

An auxiliary-input one-way function, introduced by Ostrovsky and Wigderson [OW93], is a variant of one-way function that has a weakened security condition. An auxiliary-input one-way function is defined as a collection of functions, and at least one function in the collection must be hard to invert depending on each efficient adversary. This is a natural intermediate notion between one-way functions and worst-case complexity because an adversary for an auxiliary-input one-way function must break *all* functions in the collection in the worst case, but the task of inverting each function has the average-case nature as the standard one-way function. In fact, Applebaum, Barak, and Xiao [ABX08] observed that the existence of auxiliary-input one-way function is sufficient for the hardness of PAC learning based on the same argument by Valiant [Val84]. In this section, we study the opposite direction and show that the existence of auxiliary-input one-way functions corresponds to the hardness of PAC learning under an efficiently samplable distribution over target functions (as in the BFKL model) and arbitrary unknown example distributions (as in the original PAC learning model). The main result of this section was subsumed in the work [Nan21a], which originally appeared in COLT2021, but we present a much more simplified proof in this thesis, based on the previous study [BFKL93].

Chapter 6 Learning versus Pseudorandom Generators in Constant Parallel Time

We further investigate the dichotomy between learning and cryptography in low complexity class, motivated by both (i) constructing highly efficient cryptographic primitives based on hardness assumptions of learning and (ii) identifying the capability of efficient learning for simple classes based on cryptographic assumptions. In this chapter, we establish a dichotomy between learning and polynomial-stretch pseudorandom generators (PPRGs) computable in constant parallel time (i.e., NC^0) and related classes. Note that PPRG in NC^0 is a well-studied cryptographic primitive because of its remarkable applications, such as highly efficient cryptography [IKOS08] and indistinguishability obfuscation [JLS21; JLS22], and our result is the first characterization result for such an important cryptographic primitive. More precisely, we show that the equivalence between the existence of PPRGs in NC^0 and the average-case hardness of various central classes, such as $\mathbb{F}_2$-sparse polynomials, Fourier-space functions, parity decision trees, and OR decision trees, in the BFKL model with sparse data and fixed-parameter tractable sample complexity. An important property of PPRGs in NC^0 constructed in our framework is that the output bits are computed by various predicates; thus, it seems to resist an attack that depends on a specific property of one fixed predicate. This chapter is based on the joint work with Shuichi Hirahara [HN23b], which originally appeared in ITCS2023.

Chapter 7 PACland: A World Where PAC Learning is Easy

In previous chapters, we discussed learning-theoretic aspects of Heuristica and Pessiland. In this chapter, we change the perspective and consider a possible world where PAC learning $\mathsf{P/poly}$ is easy. For familiarity, we name such a possible world *PACland* in this thesis and investigate its complexity-theoretic and cryptographic aspects. This chapter is composed of two characterization results of the hardness of PAC learning. First, we study an auxiliary-input cryptographic primitive whose existence corresponds to the hardness of PAC learning and show that PAC learning is hard if and only if there exists an auxiliary-input variant of hitting set generators with a locality condition. Second, we introduce a meta-computational problem that asks, roughly speaking, whether there

exists an efficient learner for a given concept class $\mathscr{C}$. We show that the hardness of the meta-computational problem on PAC learning itself characterizes the hardness of PAC learning. This chapter is based on the work [Nan20], which originally appeared in COLT2020.

Chapter 8 On Basing Auxiliary-Input Cryptography on NP-Hardness

The result in Chapter 7 sheds light on auxiliary-input hitting set generator whose security is weaker than the hardness of PAC learning. Particularly, basing auxiliary-input hitting set generators or other auxiliary-input cryptographic primitives on NP-hardness seems a natural intermediate step towards basing the hardness of PAC learning and cryptographic primitives on NP-hardness. Previously, Bogdanov and Trevisan [BT06b] and Akavia, Goldreich, Goldwasser, and Moshkovitz [AGGM06] presented evidence for the difficulty of basing one-way functions on NP-hardness. Particularly, they proved that such a result cannot be shown by nonadaptive black-box security reductions under the widely believed conjecture that the polynomial hierarchy does not collapse. Here, non-adaptive black-box reductions are a restrictive but still powerful form of reductions that are sufficient for many previous results shown in theoretical computer science by using reductions. We begin a study on basing auxiliary-input cryptography on NP-hardness by investigating whether such a familiar form of nonadaptive black-box security reductions suffices for proving the security of auxiliary-input cryptographic primitives. We show that if auxiliary-input hitting-set generators or auxiliary-input one-way functions are based on NP-hardness by a nonadaptive black-box security reduction, then by using the reduction repeatedly, one-way functions can also be based on NP-hardness by an *adaptive* black-box reduction, i.e., we can resolve Questions 1.0.1 and 1.1.2! This result can be interpreted from two different perspectives: optimistically, it provides new approach for Question 1.1.2 that reduces constructing a one-way function to constructing a cryptographic primitive that has much weaker security requirements, and pessimistically, basing auxiliary-input cryptographic primitives on NP-hardness by a nonadaptive black-box reduction is harder than solving the longstanding open problem of excluding Heuristica and Pessiland. In any case, our result enhances the significance of the further study of auxiliary-input cryptographic primitives. This chapter is based on the work [Nan21b], which originally appeared in ITCS2021.

Chapter 9 New and Improved Oracle Separations

We discuss the limitations of the techniques presented in this thesis. Particularly, all the main results in the previous chapters can be shown with *relativizing* proofs, i.e., the same results hold even with additional access to an arbitrary oracle. We show that, for further conceptual improvement of our results, we require profoundly new *non-relativized* ideas by presenting new strong oracle separations. Our oracle construction also provides new insights into other important topics related to average-case complexity.

Section 9.1 Worst-Case vs. Average-Case Complexity

We present a strong oracle separation between the worst-case and average-case complexity. Previously, Impagliazzo [Imp11] presented an oracle relative to which NP is easy on average in the errorless setting (i.e., $\mathsf{DistNP} \subseteq \mathsf{AvgP}$), and $\mathsf{UP} \cap \mathsf{coUP}$ ($\subseteq \mathsf{NP}$) is hard in worst case even for $2^{O(n^{1/4})}$-size circuits. First, we improve the previous oracle construction drastically by constructing an oracle relative to which PH ($\supseteq$ NP) is easy on average in the errorless setting, and $\mathsf{UP} \cap \mathsf{coUP}$ is hard in the worst case even for $2^{o(n/\log n)}$-size circuits. Note that our oracle separation is tight

because of the matching upper bound by Hirahara [Hir21a], who proved that the errorless average-case easiness of PH implies that PH (thus also $\mathsf{UP} \cap \mathsf{coUP}$) is solvable in $2^{O(n/\log n)}$ time. We further strengthen the oracle separation to provide a strong relativization barrier against removing the computational assumption on example distributions of the main theorem in Section 3.1. More precisely, we construct an oracle relative to which PH is easy on average in the errorless setting, and PAC learning linear-size circuits is infeasible even in weak learning (with a fixed accuracy parameter $\epsilon = 1/2 - 1/\mathsf{poly}(n)$) with additional membership queries by non-uniform $2^{o(n/\log n)}$-time learners under uniform example distributions over a subset $S \subseteq \{0,1\}^n$ with $|S| = 2^{(1-\alpha)n}$ for an arbitrarily small constant $\alpha > 0$. Notice that if $|S| = 2^n$ (i.e. $\alpha = 0$), then such an example distribution is uniform and trivially samplable; thus, the feasibility of PAC learning follows from the theorem in Section 3.1. This section is based on the last half of the FOCS2021 paper [HN21]. We remark that we strengthen the original result in this thesis by considering the non-uniform computation model instead of the uniform randomized computational model on the hardness part of the oracle separation.

Section 9.2 Errorless vs. Error-Prone Average-Case Complexity

We present a strong oracle separation between the errorless and error-prone average-case complexity. The question of showing such an oracle separation was first posed in the seminal paper by Impagliazzo [Imp95] and later remarked in [Imp11], but it has been open over two decades. We resolve the problem open for decades by constructing an oracle relative to which NP is easy on average in the error-prone setting (i.e., $\mathsf{DistNP} \subseteq \mathsf{HeurP}$) but hard on average in the errorless setting even by $2^{o(n/\log n)}$-size circuits. Furthermore, we construct an oracle relative to which $\mathsf{DistNP} \subseteq \mathsf{HeurP}$, but there is an auxiliary-input one-way function secure against $2^{o(n/\log n)}$-size circuits, which shows (i) strong relativization barriers against improving the main results in Chapters 3 and 4 and (ii) incomparability between auxiliary-input cryptography and average-case hardness of NP in the relativized world along with the previous work by Wee [Wee06], who presented an oracle that separates the same notions in the *opposite* way. We also remark that our oracle separation result drastically improves the previous separation result between one-way functions and auxiliary-input one-way functions [Tre10; Nan21b]. This section is based on the joint work with Shuichi Hirahara [HN22], which originally appeared in CCC2022.

Chapter 10 Conclusions and Future Directions

We conclude the thesis with several open questions on the relationships between learning, complexity theory, and cryptography.

1.3.1 List of Original Publications

As outlined above, this thesis is mainly composed of the following studies made by the author (with collaborators) during the doctoral degree program at Tokyo Institute of Technology.

[**Nan20**] M. Nanashima, Extending Learnability to Auxiliary-Input Cryptographic Primitives and Meta-PAC Learning, in proceedings of COLT2020, Chapter 7.

[**Nan21b**] M. Nanashima, On Basing Auxiliary-Input Cryptography on NP-hardness via Non-adaptive Black-Box Reductions, in proceedings of ITCS2021, Chapter 8.

[**Nan21a**] M. Nanashima, A Theory of Heuristic Learnability, in proceedings of COLT2021, Section 5.2.

[**HN21**] S. Hirahara and M. Nanashima, On Worst-Case Learning in Relativized Heuristica, in proceedings of FOCS2021, Sections 3.1 and 9.1.

[**HN22**] S. Hirahara and M. Nanashima, Finding Errorless Pessiland in Error-Prone Heuristica, in proceedings of CCC2022, Section 9.2.

[**HN23b**] S. Hirahara and M. Nanashima, Learning versus Pseudorandom Generators in Constant Parallel Time, in proceedings of ITCS2023, Chapter 6.

[**HN23a**] S. Hirahara and M. Nanashima, A Unified Theory of Average-Case Learning, manuscript, Chapter 4 and Section 5.1.

Chapter 2

Preliminaries

This chapter introduces basic notations and concepts required throughout the thesis.

2.1 General

All logarithms are base 2 unless otherwise specified. We use ε to represent an empty symbol. Note that we distinguish ε from ϵ and often use ϵ for a small real value or an accuracy parameter. For each $n \in \mathbb{N}$, we let 1^n denote a unary string $11\cdots11$ of length n. For each $n \in \mathbb{N}$, let $\{0,1\}^{\leq n} := \cup_{i=1}^{n}\{0,1\}^i$.

Let $\langle , \rangle$ be a (standard) paring function that maps $\mathbb{N}\times\mathbb{N}$ to $\mathbb{N}$. For each $k \in \mathbb{N}$ and $n_1, \ldots, n_k \in \mathbb{N}$, we use the notation $\langle n_1, \ldots, n_k\rangle$ to represent $\langle n_1, \langle n_2, \langle \cdots, \langle n_{k-1}, n_k\rangle\rangle\rangle\rangle \in \mathbb{N}$. For each $k \in \mathbb{N}$ and $x_1, \ldots, x_k \in \{0,1\}^*$, we also abuse the notation $\langle x_1, \ldots, x_k\rangle$ to represent the (standard) binary encoding for the k-tuple $(x_1, \ldots, x_k)$. For every $k, k' \in \mathbb{N}$ with $k \leq k'$, let $[k : k']$ denote a set $\{k, k+1, \ldots, k'\}$. For each $k \in \mathbb{N}$, let $[k] := [1 : k] = \{1, \ldots, k\}$.

For any $x \in \{0,1\}^*$, we let $|x|$ denote its length and let $wt(x)$ denote the Hamming weight of x. For every $x, y \in \{0,1\}^*$, let $x \circ y$ denote the concatenation of x and y. For readability, we may omit $\circ$ from $x \circ y$. For each $x \in \{0,1\}^n$ and each $i \in [n]$, we let x_i denote the i-th bit of x. For every $x \in \{0,1\}^n$ and every $S = \{i_1, \ldots, i_k\} \subseteq [n]$ (where $i_1 < \cdots < i_k$), we let x_S denote $x_{i_1} \circ \cdots \circ x_{i_k}$, particularly, $x_{[k]} = x_1 \circ \cdots \circ x_k$ and $x_{[k:k']} = x_k \circ \cdots \circ x_{k'}$ for each $k \leq k' \leq n$. For convenience, we define $x_{[k:k']}$ for any $k, k' \in \mathbb{N}$ with $k \leq k'$ as $x_{[k:k']\cap[|x|]}$; e.g., $01101_{[3:7]} = 01101_{[3:5]} = 101$ and $011_{[10:20]} = \varepsilon$.

We use the notation negl to represent some negligible function, i.e., for any polynomial p and sufficiently large $n \in \mathbb{N}$, it holds that $\mathsf{negl}(n) < 1/p(n)$. We also use the notation poly to refer to some polynomial.

For every function $f\colon \mathcal{X} \to \mathcal{Y}$ and every $y \in \mathrm{Im}f := \{f(x) : x \in \mathcal{X}\}$, let $f^{-1}(y) := \{x \in \mathcal{X} : f(x) = y\}$. For any $f\colon \{0,1\}^n \to \{0,1\}$ and any $k \in \mathbb{N}$ with $k \leq n$, we say that f is k-junta if f depends on only at most k out of n coordinates in the input. We say that a multi-output function $f\colon \{0,1\}^n \to \{0,1\}^m$ has locality k if each output bit of f is computed by a k-junta.

For any distribution $\mathcal{D}$, we use the notation $x \sim \mathcal{D}$ to refer to the sampling of x according to D. For any finite set S, we also use the notation $x \sim S$ to refer to the uniform sampling of x from S. For any distribution $\mathcal{D}$, we denote by $\mathrm{supp}(\mathcal{D})$ the support of $\mathcal{D}$. For each $n \in \mathbb{N}$, let U_n denote the uniform distribution over $\{0,1\}^n$. For simplicity, we may identify a distribution $\mathcal{D}$ with a random variable drawn from $\mathcal{D}$. For each distribution $\mathcal{D}$ and $m \in \mathbb{N}$, let $\mathcal{D}^m$ denote the distribution of

$x_1 \circ \cdots \circ x_m$, where $x_1, \ldots, x_m \sim \mathcal{D}$.

For each distribution $\mathcal{D}$ and each $x \in \{0,1\}^*$, let $\mathcal{D}(x) = \Pr_{y\sim D}[y = x]$. Furthermore, we let $\mathcal{D}^*(x)$ denote the probability that a string whose prefix matches x is selected according to $\mathcal{D}$, i.e., $\mathcal{D}^*(x) = \Pr_{y\sim\mathcal{D}}[y \text{ begins with } x]$.

For any distribution $\mathcal{D}$, let $H(\mathcal{D})$ denote the Shannon entropy of $\mathcal{D}$. For any distributions $\mathcal{D}$ and $\mathcal{E}$, let $\mathsf{L}_1(\mathcal{D},\mathcal{E})$ denote the total variation distance between $\mathcal{D}$ and $\mathcal{E}$, i.e., if $\mathcal{X} = \mathrm{supp}(\mathcal{D}) \cup \mathrm{supp}(\mathcal{E})$, then

$$\mathsf{L}_1(\mathcal{D},\mathcal{E}) := \frac{1}{2}\sum_{x\in\mathcal{X}} |\mathcal{D}(x) - \mathcal{E}(x)| = \min_{f\colon \mathcal{X}\to\{0,1\}} \left| \Pr_{x\sim\mathcal{D}}[f(x) = 1] - \Pr_{x\sim\mathcal{E}}[f(x) = 1] \right|.$$

For every distribution $\mathcal{D}$ over $\{0,1\}^*$, every $x \in \{0,1\}^*$, and $k \in \mathbb{N}$, we use the notation $\mathsf{Next}_k(\mathcal{D},x)$ to refer to the conditional distribution of the k-bit prefix of a subsequent string of x selected according to $\mathcal{D}$. If x does not match any prefix in the support of $\mathcal{D}$, we regard $\mathsf{Next}_k(\mathcal{D},x)$ as the distribution of the empty symbol ε. For example, if $\mathcal{D}$ is a uniform distribution over $\{0,1\}^{\le n} := \cup_{i\le n}\{0,1\}^i$, then for every $x \in \{0,1\}^{\le n}$ and every $k \in \mathbb{N}$, $\mathsf{Next}_k(\mathcal{D},x)$ is a uniform distribution over $\{0,1\}^{\le \min\{k,n-|x|\}}$.

For any oracles $\mathcal{O}_0$ and $\mathcal{O}_1$, we let $\mathcal{O}_0 + \mathcal{O}_1$ denote the combination, i.e., for any $b \in \{0,1\}$ and any $x \in \{0,1\}^*$, $(\mathcal{O}_0 + \mathcal{O}_1)(b \circ x) = \mathcal{O}_b(x)$. For simplicity, we identify oracle access to $\mathcal{O}_0 + \mathcal{O}_1$ with access to two separated oracles $\mathcal{O}_0$ and $\mathcal{O}_1$. For every distribution $\mathcal{D}$ over binary strings and every oracle machine M, we write $M^{\mathcal{D}}$ to mean that M is given access to the oracle that returns a sample independently and identically drawn from $\mathcal{D}$ for each access.

For two distribution families $\mathcal{D} = \{\mathcal{D}_n\}_{n\in\mathbb{N}}$ and $\mathcal{E} = \{\mathcal{E}_n\}_{n\in\mathbb{N}}$, we write $\mathcal{D} \equiv_s \mathcal{E}$ to represent that $\mathcal{D}$ and $\mathcal{E}$ are statistically indistinguishable, i.e., $\mathsf{L}_1(\mathcal{D}_n,\mathcal{E}_n) \le \mathsf{negl}(n)$ for each $n \in \mathbb{N}$.

For every randomized algorithm A using $s(n)$ random bits, for each input $x \in \{0,1\}^n$, and for each internal random string $r \in \{0,1\}^{s(n)}$ for A, we use the notation $A(x;r)$ to refer to the execution of $A(x)$ with a random tape r. When an algorithm A is given some unary parameters, we abuse the notation $A(x;1^{n_1},\ldots,1^{n_m})$ to represent the execution of $A(x)$ with unary parameters $1^{n_1},\ldots,1^{n_m}$. When A is given unary parameters and a random string simultaneously, we use the notation $A(x;1^{n_1},\ldots,1^{n_m};r)$.

In this thesis, we assume basic knowledge of probability theory, including the union bound, Markov's inequality, Jensen's inequality, and Borel–Cantelli lemma. We often use the following famous concentration inequality.

Fact 2.1.1 (Hoeffding inequality)**.** *For real values $a, b \in \mathbb{R}$, let $X_1,\ldots,X_m$ be independent and identically distributed random variables with $X_i \in [a,b]$ and $\mathbb{E}[X_i] = \mu$ for each $i \in [m]$. Then for any $\epsilon > 0$, the following inequalities hold:*

$$\Pr_{X_1,\ldots,X_m}\left[\frac{1}{m}\sum_{i=1}^m X_i - \mu \ge \epsilon\right] \le e^{-\frac{2m\epsilon^2}{(b-a)^2}} \quad \text{and} \quad \Pr_{X_1,\ldots,X_m}\left[\frac{1}{m}\sum_{i=1}^m X_i - \mu \le -\epsilon\right] \le e^{-\frac{2m\epsilon^2}{(b-a)^2}}.$$

We assume basic knowledge of computational complexity such as Turing machines, universal Turing machines, randomized Turing machines, and circuits. We also assume basic knowledge of complexity classes such as $\mathsf{P}, \mathsf{BPP}, \mathsf{P/poly}, \Sigma_i^p, \Pi_i^p, \mathsf{PH}$, $\mathsf{DTIME}, \mathsf{BPTIME}$, and SIZE. Note that we abuse the notation SIZE to refer to a concept class of circuits in context of learning, as explained in Section 2.3. We often use the term *nonuniform* computation to refer to computation by circuit families (equivalently Turing machines with advice). A reader not familiar with these notions in complexity theory is referred to the textbook by Arora and Barak [AB09].

For each $t\colon \mathbb{N} \to \mathbb{N}$, we say that a family $\mathcal{D} = \{\mathcal{D}_n\}_{n\in\mathbb{N}}$ of distributions (on binary strings) is $t(n)$-time samplable if there exists a $t(n)$-time deterministic algorithm D (called a sampling algorithm or a sampler for $\mathcal{D}$) such that, for each $n \in \mathbb{N}$, the distribution of $D(1^n, U_{t(n)})$ is statistically identical to $\mathcal{D}_n$. We say that a family $\mathcal{D} = \{\mathcal{D}_n\}_{n\in\mathbb{N}}$ of distributions is (polynomial-time) samplable if $\mathcal{D}$ is $p(n)$-samplable for some polynomial $p(n)$. Let PSAMP denote a set of all samplable distributions.

We further define $\mathsf{P/poly}$-samplable distributions. In this thesis, for convenience, we fix a universal Turing machine U when we discuss $t(n)/a(n)$-samplable distributions, and we regard $a(n)$ as a function that represents the upper bound on the program size of the sampler with respect to U. This formulation of $t(n)/a(n)$-samplable corresponds to the standard meaning of time and advice complexity up to the simulation overhead of U in time complexity and an additive constant factor (precisely, the description size of the sampler that takes advice) in advice complexity.

Definition 2.1.2 (P/poly-samplable distributions)**.** *Fix a universal Turing machine U. A distribution family $\mathcal{D} = \{\mathcal{D}_n\}_{n\in\mathbb{N}}$ is said to be $t(n)/a(n)$-samplable, where $t, a\colon \mathbb{N} \to \mathbb{N}$, if for each $n \in \mathbb{N}$, there exist a problem $\Pi \in \{0,1\}^{\leq a(n)}$ such that $\mathcal{D}_n$ is statistically identical to the distribution of the output of executing $U(\Pi, U_{t(n)})$ in $t(n)$ time. We let $\mathsf{Samp}[t(n)]/a(n)$ denote a class of $t(n)/a(n)$-samplable distributions.*

2.2 Average-Case Complexity

Here, we briefly introduce the notions in average-case complexity theory. Further backgrounds can be found in a survey by Bogdanov and Trevisan [BT06a].

We define a distributional problem as a pair of a language $L \subseteq \{0,1\}^*$ and a distribution $\mathcal{D} = \{\mathcal{D}_n\}_{n\in\mathbb{N}}$ on instances. For a standard complexity class $\mathcal{C}$ (e.g., NP), we define its average-case extension $\mathsf{Dist}\mathcal{C}$ as $\mathsf{Dist}\mathcal{C} = \{(L, \mathcal{D}) : L \in \mathcal{C}, \mathcal{D} \in \mathrm{PSAMP}\}$. For convenience, we omit the description "$= \{\mathcal{D}_n\}_{n\in\mathbb{N}}$" from $\mathcal{D} = \{\mathcal{D}_n\}_{n\in\mathbb{N}}$ when the intention is clear in context.

First, we present the *errorless* average-case easiness. We say that a distributional problem $(L, \mathcal{D})$ has an *errorless* heuristic algorithm A with failure probability $\delta\colon \mathbb{N} \to (0,1)$ if (1) A outputs $L(x)$ $(:= \mathbb{1}\{x \in L\})$ or $\perp$ (which represents "failure") for every $x \in \mathrm{supp}(\mathcal{D})$, and (2) the failure probability that $A(x; 1^n)$ outputs $\perp$ over the choice of $x \sim \mathcal{D}_n$ is bounded above by $\delta(n)$ for each $n \in \mathbb{N}$. Note that an errorless heuristic algorithm never outputs an incorrect value $\neg L(x)$ for any $x \in \mathrm{supp}(\mathcal{D})$. Then, for every $\delta\colon \mathbb{N} \to (0,1)$, we define a class $\mathsf{Avg}_\delta\mathsf{P}$ as a class of distributional problems that have a polynomial-time errorless heuristic algorithm with failure probability $\delta(n)$. Furthermore, we say that a distributional problem $(L, \mathcal{D})$ has an *errorless* heuristic scheme A if A is given an instance $x \in \mathrm{supp}(\mathcal{D}_n)$ and parameters 1^n and $1^{\delta^{-1}}$, where $n, \delta^{-1} \in \mathbb{N}$, as input and satisfies the condition of an errorless heuristic algorithm with failure probability δ. We define a class AvgP as a class of distributional problems that have a polynomial-time errorless heuristic scheme. It is not hard to verify that $\mathsf{AvgP} \subseteq \mathsf{Avg}_{1/p(n)}\mathsf{P}$ for any polynomial $p(n)$.

Next, we present the *error-prone* average-case easiness. We say that a distributional problem $(L, \mathcal{D})$ has an *error-prone* heuristic algorithm A with failure probability $\delta\colon \mathbb{N} \to (0,1)$ if the failure probability that $A(x) \neq L(x)$ over the choice of $x \sim \mathcal{D}_n$ is bounded above by $\delta(n)$ for each $n \in \mathbb{N}$. Note that an error-prone heuristic algorithm may output an incorrect value $\neg L(x)$, but the error probability is bounded above by $\delta(n)$. Then, for every $\delta\colon \mathbb{N} \to (0,1)$, we define a class $\mathsf{Heur}_\delta\mathsf{P}$ as a class of distributional problems that have a polynomial-time error-prone heuristic algorithm with failure probability $\delta(n)$. We also define an *error-prone* heuristic scheme and the class HeurP in the same manner as the errorless case.

We also define classes $\mathsf{AvgP/poly}$, $\mathsf{HeurP/poly}$, $\mathsf{AvgSIZE}[s(n)]$, and $\mathsf{HeurSIZE}[s(n)]$ for each size parameter $s(n)$ in the same manner as above. Below, we explicitly describe the average-case easiness in the randomized computational model because it is a little complicated due to the two kinds of failures caused by hard instances and bad randomness. Note that the latter failure probability is easily reduced by the standard repeating technique.

Definition 2.2.1 (Randomized heuristic scheme)**.** *A randomized algorithm A is said to be a randomized errorless heuristic scheme for a distributional problem $(L, \mathcal{D})$ if A satisfies that for any $n, \delta^{-1} \in \mathbb{N}$,*

1. *$A(x; 1^n, 1^{\delta^{-1}}) \in \{0, 1, \bot\}$ and $\Pr_A[A(x; 1^n, 1^{\delta^{-1}}) = \neg L(x)] \leq 1/4$ for any $x \in \mathrm{supp}(\mathcal{D}_n)$;*
2. *$\Pr_{x \sim \mathcal{D}_n}[\Pr_A[A(x; 1^n, 1^{\delta^{-1}}) = \bot] \geq 1/4] \leq \delta$.*

Furthermore, a randomized algorithm A is said to be a randomized heuristic scheme for a distributional problem $(L, \mathcal{D})$ if A satisfies that for any $n, \delta^{-1} \in \mathbb{N}$,

$$\Pr_{x \sim \mathcal{D}_n}\left[\Pr_A[A(x; 1^n, 1^{\delta^{-1}}) \neq L(x)] \leq 1/4\right] \leq \delta.$$

AvgBPP, $\mathsf{HeurBPP}$, $\mathsf{AvgBPTIME}[t(n)]$, and $\mathsf{HeurBPTIME}[t(n)]$ can be defined based on the definition of randomized heuristic schemes as in the deterministic computational model.

2.3 Learning Theory

A concept class is defined as a subset of Boolean-valued functions $\{f \colon \{0,1\}^n \to \{0,1\} : n \in \mathbb{N}\}$. For any concept class $\mathscr{C}$, we use the notation $\mathscr{C}_n$ to represent $\mathscr{C} \cap \{f \colon \{0,1\}^n \to \{0,1\}\}$ for each $n \in \mathbb{N}$. We assume that every concept class $\mathscr{C}$ has a binary encoding and an evaluation $C \colon \{0,1\}^* \times \{0,1\}^* \to \{0,1\}$ satisfying $C(f, x) = f(x)$ for each $n \in \mathbb{N}$, $f \in \mathscr{C}_n$, and $x \in \{0,1\}^n$, where we identify f with its binary encoding. In this thesis, we assume that every concept class is evaluatable in polynomial time in input size n unless otherwise stated, i.e., we consider concept classes that have polynomial-length encodings and polynomial-time computable evaluations. For every $\mathscr{C}$, we use the notation $\ell_{\mathscr{C}}$ to refer to a polynomial $\ell_{\mathscr{C}} \colon \mathbb{N} \to \mathbb{N}$ such that each $f \in \mathscr{C}_n$ has a binary encoding of length at most $\ell_{\mathscr{C}}(n)$. We define a dual $\mathscr{C}^*$ of a concept class $\mathscr{C}$ with the evaluation $C \colon \{0,1\}^* \times \{0,1\}^* \to \{0,1\}$ as a class determined by the evaluation $C^* \colon \{0,1\}^* \times \{0,1\}^* \to \{0,1\}$ defined as $C^*(u, x) = C(x, u)$ for each $u, x \in \{0,1\}^*$.

We define an example distribution as a distribution $\mathcal{D} = \{\mathcal{D}_n\}_{n \in \mathbb{N}}$ such that each $\mathcal{D}_n$ is over $\{0,1\}^n$. Note that we often omit the description "$= \{\mathcal{D}_n\}_{n \in \mathbb{N}}$" and implicitly assume that each $\mathcal{D}_n$ is over $\{0,1\}^n$ when we discuss an example distribution $\mathcal{D}$. We also define a class of example distributions as a set of example distributions. For any class $\mathscr{D}$ of example distributions and $n \in \mathbb{N}$, we use the notation $\mathscr{D}_n$ to represent $\mathscr{D}_n = \{\mathcal{D}_n : \mathcal{D} \in \mathscr{D}\}$.

First, we formally introduce the probably approximately correct (PAC) learning model, introduced by Valiant [Val84], and a generalized model called the agnostic learning model, introduced by Kearns, Schapire, and Sellie [KSS94]. Note that almost all learning models we discuss in this thesis are variants of PAC learning and agnostic learning models.

Definition 2.3.1 (PAC learning and agnostic learning [Val84; KSS94])**.** *Let $\mathscr{C}$ be a concept class and $\mathscr{D}$ be a class of example distributions. We say that a randomized oracle machine L, referred to*

as an agnostic learner, agnostically learns $\mathscr{C}$ *on* $\mathscr{D}$ *(with time complexity* $t\colon \mathbb{N}\times(0,1]\times(0,1]\to\mathbb{N}$ *and sample complexity* $m\colon \mathbb{N}\times(0,1]\times(0,1]\to\mathbb{N}$*) if* L *satisfies the following conditions:*

1. *L is given parameters $1^n, 1^{\epsilon^{-1}}, 1^{\delta^{-1}}$ as the input (where $n, \epsilon^{-1}, \delta^{-1} \in \mathbb{N}$, and ϵ and δ are called an accuracy parameter and a confidence parameter, respectively) and given access to an example oracle $\mathsf{EX}_{f,\mathcal{D}_n}$ determined by a (possibly randomized)[1] target function $f\colon \{0,1\}^n \to \{0,1\}$ and an example distribution $\mathcal{D}_n \in \mathscr{D}_n$.*

2. *For each access, $\mathsf{EX}_{f,\mathcal{D}_n}$ returns an example of the form $(x, f(x))$, where x is selected identically and independently according to $\mathcal{D}_n$.*

3. *For all example distributions $\mathcal{D} \in \mathscr{D}$, all large enough $n \in \mathbb{N}$, all $\epsilon^{-1}, \delta^{-1} \in \mathbb{N}$, and all randomized target functions $f\colon \{0,1\}^n \to \{0,1\}$, the learner L outputs a description of a polynomial-time-evaluatable function $h\colon \{0,1\}^n \to \{0,1\}$ as a hypothesis with probability at least $1-\delta$ so that h is ϵ-close to the best function in $\mathscr{C}$ that approximates f under $\mathcal{D}$, i.e., L satisfies the following condition:*

$$\Pr_{L,\mathsf{EX}_{f,\mathcal{D}_n}}\left[L^{\mathsf{EX}_{f,\mathcal{D}_n}}(1^n, 1^{\epsilon^{-1}}, 1^{\delta^{-1}}) \text{ outputs } h \text{ such that } \Pr_{f,x\sim\mathcal{D}_n}[h(x)\neq f(x)] \le \mathsf{opt}_{\mathscr{C},f,\mathcal{D}_n} + \epsilon\right] \ge 1-\delta,$$

where $\mathsf{opt}_{\mathscr{C},f,\mathcal{D}_n} = \min_{f^*\in\mathscr{C}_n} \Pr_{f,x\sim\mathcal{D}_n}[f^*(x)\neq f(x)]$.

4. *L halts in time $t(n,\epsilon,\delta)$ with access to the example oracle at most $m(n,\epsilon,\delta)$ times in each case.*

A PAC learner L (with time complexity $t(n,\epsilon,\delta)$ and sample complexity $m(n,\epsilon,\delta)$) on $\mathscr{D}$ is defined as a randomized oracle machine L satisfying the above-mentioned conditions 1, 2, and 4, as well as condition 3, except that we only consider the realizable case of $f \in \mathscr{C}$, i.e., $\mathsf{opt}_{\mathscr{C},f,\mathcal{D}_n} = 0$ (instead of all randomized target functions).

We say that $\mathscr{C}$ is agnostic (resp. PAC) learnable in polynomial time on $\mathscr{D}$ if there is an agnostic (resp. PAC) learner for $\mathscr{C}$ on $\mathscr{D}$ with time complexity $t(n,\epsilon,\delta) \le \mathsf{poly}(n,\epsilon^{-1},\delta^{-1})$. In addition, we say that $\mathscr{C}$ is weakly learnable on $\mathscr{D}$ if there exists a PAC learner for $\mathscr{C}$ on $\mathscr{D}$ with some fixed accuracy and confidence parameters $\epsilon := \epsilon(n) \le 1/2 - 1/\mathsf{poly}(n)$, $\delta := \delta(n) \le 1 - 1/\mathsf{poly}(n)$.

We may grant a learner additional access to another oracle MQ_f, referred to as a membership query oracle, that returns $f(x)$ for each membership query x by the learner.

For a function $s\colon \mathbb{N}\to\mathbb{N}$, we define a concept class $\mathsf{SIZE}[s(n)]$ of circuits by

$$\mathsf{SIZE}[s(n)] = \{f\colon \{0,1\}^n \to \{0,1\} \ : \ n\in\mathbb{N} \text{ and } f \text{ is computable by an } s(n)\text{-size circuit}\}.$$

Note that agnostic (resp. PAC) learning $\mathsf{SIZE}[n^2]$ is regarded as a complete problem for agnostic (resp. PAC) learning in the following sense: if $\mathsf{SIZE}[n^2]$ is agnostic (resp. PAC) learnable in polynomial time iff all polynomial-time-evaluatable classes are agnostic (resp. PAC) learnable in polynomial time by the simple padding argument.

[1] Namely, we assume that each $f(x)$ is associated with some distribution $\mathcal{D}_x$ on $\{0,1\}$, and the outcome of $f(x)$ is distributed according to $\mathcal{D}_x$.

2.3.1 Average-Case Learning: Blum–Furst–Kearns–Lipton Model

Blum, Furst, Kearns, and Lipton [BFKL93] proposed a natural average-case variant of the PAC learning model to study implications from average-case hardness of learning to cryptography. In this thesis, we call their model *the BFKL model* to explicitly distinguish it from other average-case learning tasks.

In the BFKL model, we consider the average-case setting in which a target function is selected at random according to a distribution $\mathcal{F}$ over $\mathscr{C}$. A distribution $\mathcal{F} = \{\mathcal{F}_n\}_{n\in\mathbb{N}}$, where each $\mathcal{F}_n$ is over $\mathscr{C}_n$, is said to be samplable if there exists a polynomial-time sampling algorithm F such that F outputs a binary representation of functions in $\mathscr{C}$, and for each $n \in \mathbb{N}$, the distribution of $F(1^n, r)$ (where r is a random seed) corresponds to $\mathcal{F}_n$. For simplicity, we omit the description "$= \{\mathcal{F}_n\}_{n\in\mathbb{N}}$" and implicitly assume that each $\mathcal{F}_n$ is over $\mathscr{C}_n$ when we define a distribution $\mathcal{F}$ over a concept class $\mathscr{C}$.

Definition 2.3.2 (Blum–Furst–Kearns–Lipton (BFKL) model)**.** *Let $\mathscr{C}$ be a concept class, let $\mathcal{F}$ be a distribution over $\mathscr{C}$, and let $\mathcal{D}$ be an example distribution. We say that $\mathscr{C}$ is PAC learnable on average in the BFKL model with respect to $\mathcal{D}$ and $\mathcal{F}$ if there exists a randomized oracle machine $L^?$ (i.e., a learner) such that for every large enough $n \in \mathbb{N}$ and every $\epsilon^{-1}, \delta^{-1} \in \mathbb{N}$,*

$$\Pr_{f\sim\mathcal{F}_n, \mathsf{EX}_{f,\mathcal{D}_n}, L}\left[L^{\mathsf{EX}_{f,\mathcal{D}_n}}(1^n, 1^{\epsilon^{-1}}, 1^{\delta^{-1}}) \text{ outputs } h \text{ such that } \Pr_{x\sim\mathcal{D}_n}[h(x) \neq f(x)] \leq \epsilon\right] \geq 1-\delta.$$

We also define weak learning in the BFKL model, where the parameters ϵ and δ are fixed to be $\epsilon := \epsilon(n) \leq 1/2 - 1/\mathsf{poly}(n)$ and $\delta := \delta(n) \leq 1 - 1/\mathsf{poly}(n)$.

Blum, Furst, Kearns, and Lipton [BFKL93] (along with [Val84; GGM86; HILL99]) proved that the average-case hardness in their model characterizes the existence of one-way functions (see Section 2.5 for the formal definition of one-way functions).

Theorem 2.3.3 ([BFKL93])**.** *The following are equivalent.*

1. *There exists an infinitely-often one-way function.*
2. *There exist a polynomial-time evaluatable concept class $\mathscr{C}$, a samplable distribution $\mathcal{F}$ over $\mathscr{C}$, and a samplable example distribution $\mathcal{D}$ such that $\mathscr{C}$ is not PAC learnable in polynomial time on average in the BFKL model with respect to $\mathcal{D}$ and $\mathcal{F}$.*
3. *There exist a polynomial-time evaluatable concept class $\mathscr{C}$ and a samplable distribution $\mathcal{F}$ over $\mathscr{C}$ such that $\mathscr{C}$ is not weekly learnable in polynomial time on average in the BFKL model with respect to the uniform example distribution and $\mathcal{F}$.*

Strictly speaking, the original formulation of average-case weak learning in [BFKL93] is based on the prediction model, introduced in [PW90], rather than the PAC learning model. These two formulations are equivalent by a similar proof as in [HKLW88]. In this thesis, we also use the original formulation based on the prediction model. For completeness, we prove the equivalence between weak learning and weak prediction below.

Definition 2.3.4 (Weak prediction in BFKL model)**.** *Let $\mathscr{C}$ be a concept class, let $\mathcal{F}$ be a distribution over $\mathscr{C}$, and let $\mathcal{D}$ be an example distribution. We say that $\mathscr{C}$ is weakly predictable on average*

in the BFKL model with respect to $\mathcal{D}$ and $\mathcal{F}$ if there exist polynomials $m := m(n)$ and $p(n)$ and a randomized algorithm P such that for all large enough $n \in \mathbb{N}$,

$$\Pr_{f\sim\mathcal{F}_n, P, x^1,\dots,x^m,x^*\sim\mathcal{D}_n}\left[P\left(\left(x^1, f(x^1)\right),\dots,\left(x^m, f(x^m)\right), x^*\right) = f(x^*)\right] \geq \frac{1}{2} + \frac{1}{p(n)}.$$

The function $1/p(n)$ above is called advantage of P, and x^* is called a challenge.

Theorem 2.3.5. *For any concept class $\mathscr{C}$, any distribution $\mathcal{F}$ over $\mathscr{C}$, and any example distribution, the following statements are equivalent in the BFKL model with respect to $\mathcal{D}$ and $\mathcal{F}$:*

1. *$\mathscr{C}$ is weakly predictable in polynomial time on average.*
2. *$\mathscr{C}$ is weakly learnable in polynomial time for fixed parameters $\epsilon := \epsilon(n)$ and $\delta := \delta(n)$ such that*
$$\epsilon \leq \frac{1}{2} - \frac{1}{p_\epsilon(n)} \text{ and } \delta \leq 1 - \frac{1}{p_\delta(n)},$$
where p_ϵ and p_δ are some polynomials.

Proof. (1 ⇒ 2) Let P be a prediction algorithm for $\mathscr{C}$ of advantage $1/p(n)$. Then we construct a weak learner L as follows: on input 1^n, the learner L outputs a hypothesis h defined as

$$h_{S,r}(x) = P\left(S, x; r\right),$$

where S denotes a sample set required for executing P (notice that L can obtain a sample from EX), and r denotes randomness for P selected by L. Obviously, L halts in polynomial time in n.

Since L executes P in the valid settings for the same target function f, we have

$$\Pr_{f\sim\mathcal{F}_n, r, S, x\sim\mathcal{D}_n}\left[h_{S,r}(x) \neq f(x)\right] \leq \frac{1}{2} - \frac{1}{p(n)}.$$

By Markov's inequality,

$$\Pr_{f,r,S}\left[\Pr_{x\sim\mathcal{D}_n}\left[h_{S,r}(x) \neq f(x)\right] \geq \frac{1}{2} - \frac{1}{2p(n)}\right] \leq \frac{\frac{1}{2} - \frac{1}{p(n)}}{\frac{1}{2} - \frac{1}{2p(n)}} = 1 - \frac{1}{p(n) - 1} \leq 1 - \frac{1}{p(n)}.$$

Therefore, L achieves the parameters $\epsilon = 1/2 - 1/2p(n)$ and $\delta = 1 - 1/p(n)$.

(2 ⇒ 1) Let L be a weak learner for $\mathscr{C}$ with parameters ϵ and δ as in Theorem 2.3.5. First, we apply the standard repeating and testing technique to reduce the failure probability of L over the choice of samples and randomness for L to 2^{-n} with multiplicative loss of time $\mathsf{poly}(n)$ [cf. HKLW88, Lemma 3.4]. For simplicity, we assume that L never fails over the choice of samples and randomness for L, which is valid because the failure probability 2^{-n} affects the success probability only negligibly.

We construct a weak predictor P for $\mathscr{C}$ that takes large enough samples to successfully execute $L^?(1^n)$. If L outputs some hypothesis h, then P estimates the probability p_h that h agrees with the target function under the example distribution within accuracy $\pm 1/2p_\epsilon(n)$ with probability at least $1 - 2^{-n}$. By the standard empirical estimation, it is enough to take $M = O(p_\epsilon(n)^2 n) = \mathsf{poly}(n)$ samples for estimating p_h, and again we can assume that the failure probability is 0 instead of 2^{-n}. If the estimate $\tilde{p}$ is at least $1/2 + 1/2p_\epsilon(n)$, then P outputs $h(x^*)$, otherwise outputs a random bit

as P's prediction. It can be easily verified that the number of required samples is at most $\mathsf{poly}(n)$, and P halts in time $\mathsf{poly}(n)$.

Notice that L always succeeds in learning with probability at least $1-\delta$ over the choice of target functions and succeeds in estimating p_h within error $\pm 1/2p_\epsilon(n)$ with probability 1 (because of the assumptions above). Let f denote a target function. If L outputs a hypothesis h that is ϵ-close to f, then the estimate $\tilde{p}$ satisfies the condition $\tilde{p} \geq (1-\epsilon) - 1/2p_\epsilon(n) \geq 1/2 + 1/2p_\epsilon(n)$, and h is used for prediction. On the other hand, if a hypothesis h passes the test, then h must satisfy the condition $p_h + 1/2p_\epsilon(n) \geq 1/2 + 1/2p_\epsilon(n)$, i.e., $p_h \geq 1/2$. Even if h does not pass the test, P makes a prediction at random. Thus, for any target function, P succeeds in predicting with probability at least $1/2$. Therefore, P's success probability is at least

$$\begin{aligned}(1-\delta)(1-\epsilon) + \delta \cdot \frac{1}{2} &\geq (1-\delta)\left(\frac{1}{2} + \frac{1}{p_\epsilon(n)}\right) + \delta \cdot \frac{1}{2} \\ &= \frac{1}{2} + \frac{1-\delta}{p_\epsilon(n)} \\ &\geq \frac{1}{2} + \frac{1}{p_\delta(n)p_\epsilon(n)}.\end{aligned}$$

□

2.3.2 Random-Right-Hand-Side Refutation

Next, we introduce a recent characterization of PAC learnability and agnostic learnability by another task called *refutation*, which was studied independently by Vadhan [Vad17] and Kothari and Livni [KL18].

First, we present the notion of random-right-hand-side (RRHS) refutation introduced by Vadhan [Vad17]. Intuitively, RRHS refutation for a concept class $\mathscr{C}$ is a task of distinguishing sample sets that have uniformly random labels from sample sets realizable by $\mathscr{C}$ with one-sided error.

Definition 2.3.6 (RRHS refutation)**.** *Let $\mathscr{C}$ be a concept class. We say that a randomized algorithm A, called a refuter, RRHS-refutes $\mathscr{C}$ with $m := m(n)$ samples if for any large enough $n \in \mathbb{N}$, any $f \in \mathscr{C}_n$, and any $x_1, \ldots, x_m \in \{0,1\}^n$, the refuter A satisfies that*

1. *Soundness:* $\Pr_A[A(x_1, \ldots, x_m, f(x_1), \ldots, f(x_m)) = \text{“realizable”}] \geq 2/3$;
2. *Completeness:* $\Pr_{b\sim\{0,1\}^m}[\Pr_A[A(x_1, \ldots, x_m, b) = \text{“random”}] \geq 2/3] > 1/2$.

We say that $\mathscr{C}$ is RRHS-refutable (in polynomial time) if there exist a polynomial m and a polynomial-time refuter for $\mathscr{C}$ with m samples.

Vadhan [Vad17] proved the equivalence between PAC learning and RRHS refutation. The high-level idea is to use Yao's next-bit generator [Yao82] and boosting [Sch90].

Theorem 2.3.7 ([Vad17])**.** *A concept class $\mathscr{C}$ is PAC learnable in polynomial time if and only if $\mathscr{C}$ is RRHS-refutable in polynomial time.*

We also define a generalization of RRHS refutation introduced by Kothari and Livni [KL18], where we use a different term *correlative RRHS refutation* to refer to the term *refutation* in the original paper in order to distinguish it from RRHS-refutation and other refuting tasks for random CSPs.

For a randomized function $f\colon \{0,1\}^n \to \{0,1\}$, a concept class $\mathscr{C}$, and a distribution $\mathcal{D}$ on $\{0,1\}^n$, we define a correlation $\mathrm{Cor}_{\mathcal{D}}(f,\mathscr{C}) \in [-1,1]$ between f and $\mathscr{C}$ with respect to $\mathcal{D}$ by

$$\mathrm{Cor}_{\mathcal{D}}(f,\mathscr{C}) := \max_{c\in\mathscr{C}_n} \mathop{\mathbb{E}}_{f,x\sim\mathcal{D}}\left[(-1)^{f(x)}\cdot(-1)^{c(x)}\right] = 2\cdot\max_{c\in\mathscr{C}_n}\Pr_{f,x\sim\mathcal{D}}[f(x)=c(x)] - 1.$$

In *correlative* RRHS-refutation, a refuter is asked to distinguish sample sets that have uniformly random labels from sample sets that have correlation with a concept class rather than realizable ones.

Definition 2.3.8 (Correlative RRHS refutation)**.** *Let $\mathscr{C}$ be a concept class, $m\colon \mathbb{N}\times(0,1]\to\mathbb{N}$ be a function, and $\mathcal{D}$ be an example distribution. We say that a randomized algorithm A correlatively RRHS-refutes $\mathscr{C}$ with sample complexity m on $\mathcal{D}$ if for any large enough $n\in\mathbb{N}$ and any $\epsilon^{-1}\in\mathbb{N}$, the refuter A satisfies the following conditions:*

1. *Soundness: if samples $S=\{(x^{(i)},b^{(i)})\}_{i=1}^m$ are selected identically and independently according to $\mathsf{EX}_{f,\mathcal{D}_n}$ for a randomized function f such that $\mathrm{Cor}_{\mathcal{D}_n}(f,\mathscr{C})\geq\epsilon$, then*

$$\Pr_{S,A}[A(1^n,1^{\epsilon^{-1}},S) \text{ outputs ``correlative''}] \geq 2/3;$$

2. *Completeness: if samples $S=\{(x^i,b^i)\}_{i=1}^m$ are selected identically and independently according to $\mathsf{EX}_{f_R,\mathcal{D}_n}$ for a truly random function f_R (i.e., each label is selected uniformly at random), then*

$$\Pr_{S,A}[A(1^n,1^{\epsilon^{-1}},S) \text{ outputs ``random''}] \geq 2/3.$$

We say that $\mathscr{C}$ is correlatively RRHS-refutable with sample complexity m on $\mathcal{D}$ if there exists a randomized algorithm that correlatively RRHS-refutes $\mathscr{C}$ with sample complexity m on $\mathcal{D}$. We also define correlative RRHS refutation on a class of example distributions in the natural manner.

Kothari and Livni [KL18] proved that correlative RRHS refutation characterizes agnostic learning.

Theorem 2.3.9 ([KL18])**.** *Let $\mathscr{C}$ be a concept class, and let $\mathcal{D}$ be an example distribution. If $\mathscr{C}$ is correlatively RRHS-refutable on $\mathcal{D}$ with $m(n,\epsilon)$ samples in time $T(n,\epsilon)$, then $\mathscr{C}$ is agnostic learnable on $\mathcal{D}$ with sample complexity $O(\frac{m(n,\epsilon/2)^3}{\epsilon^2})$ and time complexity $O(T(n,\epsilon/2)\cdot\frac{m(n,\epsilon/2)^2}{\epsilon^2})$. Furthermore, the same result holds for a class of example distributions.*

We remark that the characterization of agnostic learning by correlative RRHS refutation holds in distribution-specific cases because of distribution-specific boosting in agnostic learning [Fel10]. In fact, the usage of distribution-specific boosting is essential to obtain the main result in Section 3.1 because a standard boosting technique [e.g., Sch90; FS96] uses a weak learner many times on various example distributions adaptively depending on weak hypotheses in previous stages, but we consider a weak learner whose complexity depends on the computational complexity for sampling according to the example distribution. Namely, the standard boosting technique causes a super-polynomial blowup of time complexity in our case.

2.4 Meta-Complexity and Kolmogorov Complexity

Meta-complexity refers to the field that studies the complexity of determining complexity and has deep connections to learning theory [e.g., CIKK16; CIKK17; OS17; San20; ILO20; Hir22a]. We introduce several important notions in meta-complexity we require in this thesis.

First, we introduce one of the central complexity notions called Kolmogorov complexity. Intuitively, the t-time-bounded Kolmogorov complexity of a string x is the minimum size of a problem that prints x in t time. In this thesis, we fix a universal Turing machine U with polynomial simulation overhead.

Definition 2.4.1 (Kolmogorov complexity)**.** *For every $t \in \mathbb{N}$ and every $x, z \in \{0,1\}^*$, we define the t-time-bounded Kolmogorov complexity of x given z as*

$$\mathrm{K}^t(x|z) = \min_{p \in \{0,1\}^*} \{|p| : U^t(p, z) = x\}.$$

We also define the (time-unbounded) Kolmogorov complexity of x given z as $\mathrm{K}(x|z) = \lim_{t\to\infty} \mathrm{K}^t(x|z)$. We omit the description "$|z$" if z is the empty string, i.e., $\mathrm{K}^t(x) = \min_{p\in\{0,1\}^*}\{|p| : U^t(p) = x\}$.

For every $x \in \{0,1\}^*$, it holds that $\mathrm{K}(x) \leq |x| + O(1)$ because there exists a trivial machine M_x that outputs the embedded string x.

We use the following inequality called weak symmetry of information that holds under the average-case errorless easiness of NP. Note that further improvement of weak symmetry of information was recently obtained by Hirahara [Hir22b], but the following statement suffices for our purpose.

Theorem 2.4.2 (Weak symmetry of information [Hir21b])**.** *If $\mathsf{DistNP} \subseteq \mathsf{AvgP}$, then there exist polynomials p_0 and p_w that, for any $n, m \in \mathbb{N}$, $t \geq p_0(nm)$, $\epsilon \in (0,1]$, and $x \in \{0,1\}^n$, satisfy*

$$\Pr_{r\sim\{0,1\}^m}\left[\mathrm{K}^t(x \circ r) \geq \mathrm{K}^{p_w(t/\epsilon)}(x) + m - \log p_w(t/\epsilon)\right] \leq \epsilon.$$

We define a problem GapMINKT that asks an approximation of $\mathrm{K}^t(x)$ for given $x \in \{0,1\}^*$ and $t \in \mathbb{N}$, which is one of the most central problems in meta-complexity theory.

Definition 2.4.3 (GapMINKT)**.** *For functions $\sigma\colon \mathbb{N}\times\mathbb{N} \to \mathbb{N}$ and $\tau\colon \mathbb{N} \to \mathbb{N}$, $\mathrm{Gap}_{\sigma,\tau}\mathrm{MINKT}$ is a promise problem (Π_Y, Π_N) defined as follows:*

$$\Pi_Y = \left\{(x, 1^s, 1^t) : \mathrm{K}^t(x) \leq s\right\},$$
$$\Pi_N = \left\{(x, 1^s, 1^t) : \mathrm{K}^{\tau(|x|+t)}(x) > s + \sigma(s, |x|+t)\right\}.$$

We omit the subscript σ of $\mathrm{Gap}_{\sigma,\tau}\mathrm{MINKT}$ when $\sigma(s, |x|+t) = \log \tau(|x|+t)$.

Hirahara [Hir20b] showed that $\mathrm{Gap}_\tau\mathrm{MINKT}$ is efficiently solvable if $\mathsf{DistNP} \subseteq \mathsf{AvgP}$.

Theorem 2.4.4 ([Hir20b])**.** *If $\mathsf{DistNP} \subseteq \mathsf{AvgP}$, then $\mathrm{Gap}_\tau\mathrm{MINKT} \in \text{pr-}\mathsf{P}$ for some polynomial τ.*

Every algorithm A that solves $\mathrm{Gap}_\tau\mathrm{MINKT}$ yields the approximation algorithm $\mathsf{ApproxK}_\tau$ simply as follows. On input $x \in \{0,1\}^*$ and 1^t, where $t \in \mathbb{N}$, $\mathsf{ApproxK}_\tau$ outputs the minimum $s \in \mathbb{N}$ such that $A(x, 1^s, 1^t) = 1$. Since $(x, 1^{s-1}, 1^t)$ is not a YES instance and $(x, 1^s, 1^t)$ is not a NO instance for such s, the following lemma is easily verified.

Lemma 2.4.5. *If* $\mathrm{Gap}_\tau\mathrm{MINKT} \in \text{pr-}\mathsf{P}$*, then there exists an algorithm* $\mathsf{ApproxK}_\tau$ *that is given* $(x, 1^t)$*, where* $x \in \{0,1\}^*, t \in \mathbb{N}$*, and outputs an integer* $s \in \mathbb{N}$ *in polynomial time to satisfy*

$$\mathrm{K}^{\tau(|x|+t)}(x) - \log \tau(|x|+t) \leq s \leq \mathrm{K}^t(x).$$

Another central problem in meta-complexity is the minimum circuit size problem (MCSP) that asks the minimum size of a circuit whose truth table corresponds to a given string. Particularly, the approximation version of MCSP is stated as follows.

Definition 2.4.6 (Circuit complexity, GapMCSP)**.** *For each* $n \in \mathbb{N}$ *and* $x \in \{0,1\}^{2^n}$*, we define the circuit complexity* $\mathsf{cc}(x)$ *of* x *as the minimum size of an* n*-input circuit whose truth table corresponds to* x*.*

For a constant $\epsilon \in [0,1]$*,* $\mathrm{Gap}_\epsilon\mathrm{MCSP}$ *is a promise problem* (Π_Y, Π_N) *defined as* $\Pi_Y = \{(x, 1^s) : n \in \mathbb{N}, x \in \{0,1\}^{2^n}, \mathsf{cc}(x) \leq s\}$ *and* $\Pi_N = \{(x, 1^s) : n \in \mathbb{N}, x \in \{0,1\}^{2^n}, \mathsf{cc}(x) > 2^{(1-\epsilon)n} \cdot s\}$*.*

2.5 Cryptography

We formally introduce cryptographic primitives discussed in this thesis. Let $\mathcal{C}$ be a class of adversaries (e.g., polynomial-time randomized Turing machines and subexponential-size circuits). We regard the complexity parameter (e.g., time and size) on $\mathcal{C}$ as a function in the size of a hidden seed (referred as a security parameter) for primitives.

In this thesis, we mainly discuss cryptographic primitives with *infinitely-often* security to focus on algorithmic aspects, particularly learnability for all large enough example sizes and all parameters, where cryptographic primitives with infinitely-often security are guaranteed to be secure for infinitely many security parameters. In context of cryptography, however, cryptographic primitives with *sufficiently large* security are often considered, where such primitives are guaranteed to be secure for all sufficiently large security parameters. Our results on relationships between learnability and cryptographic primitives with infinitely-often security also hold with sufficiently large security by considering the learnability on infinitely many sample sizes n with arbitrarily small parameters fixed beforehand as polynomial-time-computable functions in n. This follows from the observation that, for each sample size n, the number of relevant security parameters is at most $\mathsf{poly}(n)$ in our reductions, and all of them are efficiently computable and bounded by a polynomial in n and the parameters (when the parameters are efficiently computable from n) and due to the standard combining trick (cf. [NR06, Lemma 4.1]).

Now, we present the formal definitions of one-way functions, pseudorandom generators, and pseudorandom functions with *infinitely-often* security. For further backgrounds of these cryptographic primitives, refer to the textbook by Goldreich [Gol01; Gol04].

Definition 2.5.1 (Infinitely-often one-way function)**.** *A polynomial-time-computable family* $f = \{f_n \colon \{0,1\}^{s(n)} \to \{0,1\}^{t(n)}\}_{n\in\mathbb{N}}$ *of functions is said to be an* infinitely-often one-way function *secure against* $\mathcal{C}$ *if, for every adversary* A *in* $\mathcal{C}$ *and every polynomial* p*, for infinitely many* $n \in \mathbb{N}$*,*

$$\Pr_{x\sim\{0,1\}^{s(n)},A}[A(1^n, f_n(x)) \in f_n^{-1}(f_n(x))] \leq 1/p(n),$$

where the probability is taken over $x \sim \{0,1\}^{s(n)}$ *and the internal randomness of* A *(if any).*

Definition 2.5.2 (Infinitely-often pseudorandom generators)**.** *A polynomial-time-computable family $G = \{G_n \colon \{0,1\}^n \to \{0,1\}^{m(n)}\}_{n\in\mathbb{N}}$ of functions is said to be an* infinitely-often pseudorandom generator *secure against $\mathcal{C}$ if, $m(n) > n$ for each $n \in \mathbb{N}$ (i.e., stretching), and for every adversary A in $\mathcal{C}$ and every polynomial p, for infinitely many $n \in \mathbb{N}$,*

$$\left| \Pr_{x\sim\{0,1\}^n, A}[A(1^n, G_n(x)) = 1] - \Pr_{w\sim\{0,1\}^{m(n)}, A}[A(1^n, w) = 1] \right| \leq 1/p(n).$$

Definition 2.5.3 (Infinitely-often pseudorandom function)**.** *A polynomial-time-computable family $f = \{f_n \colon \{0,1\}^{s(n)} \times \{0,1\}^n \to \{0,1\}^n\}_{n\in\mathbb{N}}$ is said to be an infinitely-often pseudorandom function secure against $\mathcal{C}$ if for every (oracle machine analogue of) $A^?$ in $\mathcal{C}$ and every polynomial p, for infinitely many $n \in \mathbb{N}$,*

$$\left| \Pr_{A, r\sim\{0,1\}^{s(n)}} \left[A^{f_n(r,\cdot)}(1^n) = 1\right] - \Pr_{A, \phi_n\sim F_n} \left[A^{\phi_n(\cdot)}(1^n) = 1\right] \right| \leq 1/p(n),$$

where F_n is a set of all functions that maps n-bit strings to n-bit strings.

We consider polynomial-time randomized algorithms as a class $\mathcal{C}$ of adversaries unless otherwise stated and omit the description "secure against $\mathcal{C}$" in the default case.

The following is the well-known result in cryptography.

Theorem 2.5.4 ([GGM86; HILL99])**.** *The existences of infinitely-often one-way functions, infinitely-often pseudorandom generators, and infinitely-often pseudorandom functions are equivalent.*

Next, we introduce auxiliary-input variants of cryptographic primitives, which were introduced by Ostrovsky and Wigderson [OW93]. Roughly speaking, auxiliary-input primitives are defined as a collection of candidates for secure primitives indexed by auxiliary input $z \in \{0,1\}^*$ and have a relaxed security condition that for each adversary A, there exists an auxiliary input $z_A \in \{0,1\}^*$ depending on A such that the primitive indexed by z_A is secure for A. Auxiliary-input primitives are also regarded as a generalization of infinitely-often primitives to *binary* security parameters. From the algorithmic perspective, an adversary for auxiliary-input primitives must break all primitives in the collection simultaneously, i.e., in the worst case with respect to auxiliary inputs. In this sense, auxiliary-input primitives are a natural notion sandwiched by standard cryptographic primitives and the worst-case hardness.

Definition 2.5.5 (Auxiliary-input one-way function)**.** *Let $n, m \colon \mathbb{N} \to \mathbb{N}$ be polynomials. We say that $f = \{f_z \colon \{0,1\}^{n(|z|)} \to \{0,1\}^{m(|z|)}\}_{z\in\{0,1\}^*}$ is an auxiliary-input one-way function (AIOWF) secure against $\mathcal{C}$ if each $f_z(x)$ is polynomial-time computable from (z, x), and for any adversary A in $\mathcal{C}$, there exists an infinite subset $Z_A \subseteq \{0,1\}^*$ such that for every $z \in Z_A$,*

$$\Pr\left[f_z(A(z, f_z(U_{n(|z|)}))) = f_z(U_{n(|z|)})\right] < \mathsf{negl}(|z|).$$

Definition 2.5.6 (Auxiliary-input pseudorandom generator)**.** *Let $n, m \colon \mathbb{N} \to \mathbb{N}$ be polynomials. We say that $G = \{G_z \colon \{0,1\}^{n(|z|)} \to \{0,1\}^{m(|z|)}\}_{z\in\{0,1\}^*}$ is an auxiliary-input pseudorandom generator (AIPRG) secure against $\mathcal{C}$ if each $G_z(x)$ is polynomial-time computable from (z, x), $n(\ell) < m(\ell)$ holds for any $\ell \in \mathbb{N}$, and for any adversary A in $\mathcal{C}$, there exists an infinite subset $Z_A \subseteq \{0,1\}^*$ such that for every $z \in Z_A$,*

$$\left| \Pr_{A, r\sim\{0,1\}^{n(|z|)}} [A(z, G_z(r)) = 1] - \Pr_{z\sim\{0,1\}^{m(|z|)}} [A(z, w) = 1] \right| < \mathsf{negl}(|z|).$$

Definition 2.5.7 (Auxiliary-input pseudorandom function)**.** *We say that* $f = \{f_z \colon \{0,1\}^{|z|} \times \{0,1\}^{|z|} \to \{0,1\}^{|z|}\}_{z\in\{0,1\}^*}$ *is an auxiliary-input pseudorandom function (AIPRF) secure against* $\mathcal{C}$ *if each* f_z *is polynomial-time computable from* z *and its input, and for any adversary* $A^?$ *in (an oracle machine analog of)* $\mathcal{C}$*, there exists an infinite subset* $Z_A \subseteq \{0,1\}^*$ *such that for every* $z \in Z_A$*,*

$$\left| \Pr_{A,u\sim\{0,1\}^{|z|}} \left[A^{f_z(u,\cdot)}(z) = 1\right] - \Pr_{A,\phi_{|z|}\sim F_{|z|}} \left[A^{\phi_{|z|}(\cdot)}(z) = 1\right] \right| < \mathsf{negl}(|z|),$$

where F_n *is a set of all functions that maps* n*-bit strings to* n*-bit strings.*

When the auxiliary-input z is obvious in context, we write $n(|z|)$ and $m(|z|)$ as n and m, respectively. Again, we consider polynomial-time randomized algorithms as a class $\mathcal{C}$ of adversaries for auxiliary-input cryptographic primitives unless otherwise stated and omit the description "secure against $\mathcal{C}$" in the default case.

If there exists a secure cryptographic primitive, then its auxiliary-input variant trivially exists. Note that Theorem 2.5.4 also holds for auxiliary-input cryptographic primitives.

Theorem 2.5.8 ([GGM86; HILL99])**.** *The existences of AIOWFs, AIPRGs, and AIPRFs are equivalent.*

Valiant [Val84] and Applebaum, Barak, and Xiao [ABX08] observed that the existence of AIOWF implies the hardness of PAC learning.

Theorem 2.5.9 ([Val84; ABX08])**.** *If there exists an AIOWF, then* $\mathsf{P/poly}$ *is not PAC learnable in polynomial time.*

We further introduce a hitting set generator (HSG), first introduced by Andreev, Clementi, and Rolim [ACR98], which is a pseudorandom generator with a weakened security condition that no adversary distinguishes the outcome of the generator from random strings with *one-sided* error.

Definition 2.5.10 (Hitting set generator)**.** *Let* $\ell, m \colon \mathbb{N} \to \mathbb{N}$ *be polynomials. We say that* $G = \{G_n\}_{n\in\mathbb{N}}$*, where* $G_n \colon \{0,1\}^{\ell(n)} \to \{0,1\}^{m(n)}$ *is a hitting set generator (HSG) secure against* $\mathcal{C}$ *if* G *is polynomial-time computable,* $\ell(n) < m(n)$ *holds for each* $n \in \mathbb{N}$*, and* G *hits any language recognized by adversaries in* $\mathcal{C}$ *in the following sense: For any adversary* A *in* $\mathcal{C}$*, let* $L_A = \{L_{A,n}\}_{n\in\mathbb{N}}$ *be the language recognized by* A*, where* $L_{A,n} \subseteq \{0,1\}^n$ *for each* $n \in \mathbb{N}$*. Then, for infinitely many* $n \in \mathbb{N}$*, the following holds:*

$$|L_{A,m(n)}| > \gamma \cdot 2^{m(n)} \Longrightarrow L_{A,m(n)} \cap \mathrm{Im} G_n \neq \emptyset,$$

where $\gamma = 1/2$ *(unless otherwise stated), and* γ *is called a largeness parameter.*

We define the stretch of HSG $G \colon \{0,1\}^{\ell(n)} \to \{0,1\}^{m(n)}$ as a function $\ell' \colon \mathbb{N} \to \mathbb{N}$ such that $\ell'(\ell(n)) = m(n)$ for each $n \in \mathbb{N}$.

2.6 Learning in Algorithmica (A Brief Survey)

Algorithmica is the dream world in algorithm theory, where we can efficiently solve *all* NP problems. In Algorithmica, we can accomplish many tasks we want to solve with a computer, particularly almost all types of optimization problems, such as scheduling under many constraints, finding the shortest route that visits each customer, finding the mathematically simplest proof for a given

tautology, finding new medicine that matches individuals, and so on. For further insights into Algorithmica, we recommend the reader to consult the book written by Fortnow [For13].

Impagliazzo [Imp95] also mentioned a learning-theoretic consequence in Algorithmica.

> "Less obviously, $\mathsf{P} = \mathsf{NP}$ would make trivial many aspects of the artificial intelligence program that are in real life challenging to the point of despair. Inductive learning systems would replace our feeble attempts at expert systems. One could use an *Occam's Razor* based inductive learning algorithm to automatically train a computer to perform any task that humans can. ..."

This section gives a brief survey on which learning tasks are efficiently solvable in Algorithmica. A purpose for this is to provide a clear comparison with our learnability results in *Heuristica* and *Pessiland*, where we make weaker algorithmic assumptions, instead, obtain weaker learning-theoretic consequences. Particularly, we survey NP-complete learning problems for making our final goal clear, i.e., which learning problems must be solved in Heuristica and Pessiland for excluding them from the perspective of learning theory.

Occam's Razor

First, we introduce a characterization of PAC learnability based on Occam's razor by Blumer, Ehrenfeucht, Haussler, and Warmuth [BEHW87], which yields a reduction from PAC learning to an NP-search problem.

Occam's razor is a philosophical principle that suggests choosing the simplest description for explaining phenomena well. In the context of PAC learning, Occam's razor suggests that the learner should choose the simplest hypothesis that is consistent with the given sample set. Blumer, Ehrenfeucht, Haussler, and Warmuth [BEHW87] demonstrated the validity of this approach even in the relaxed sense that compressing the given sample set sub-linearly by a consistent hypothesis suffices for PAC learning.

Theorem 2.6.1 (Occam's razor $\Rightarrow$ PAC learning [BEHW87])**.** *Let $\mathscr{C}$ be a concept class. Suppose that there exist a polynomial-time algorithm A (referred to as an Occam algorithm) and a constant $\alpha > 0$ such that for every $n, m \in \mathbb{N}$, every $x^1, \ldots, x^m \in \{0,1\}^n$, and every $f \in \mathscr{C}_n$, the algorithm A is given $(x^1, f(x^1)), \ldots, (x^m, f(x^m))$ and outputs a hypothesis $h\colon \{0,1\}^n \to \{0,1\}$ with probability at least $2/3$ such that (i) $h(x^i) = f(x^i)$ for each $i \in [m]$ (i.e., consistent) and (ii) h is encoded by $\mathsf{poly}(n) \cdot m^{1-\alpha}$ bits. Then, A is also a polynomial-time PAC learner for $\mathscr{C}$ for sufficiently large sample complexity $m(n, \epsilon, \delta) = \mathsf{poly}(n, \epsilon^{-1}, \delta^{-1})$, and $\mathscr{C}$ is thus PAC learnable in polynomial time.*

Furthermore, Schapire [Sch90] proved the converse of the result above. Therefore, PAC learning is indeed characterized by finding a consistent hypothesis that compresses the given sample set.

Theorem 2.6.2 (PAC learning $\Rightarrow$ Occam's razor [Sch90])**.** *If a concept class $\mathscr{C}$ is PAC learnable in polynomial time, then there exists a polynomial-time Occam algorithm for $\mathscr{C}$.*

Notice that the task of finding a consistent hypothesis is an NP-search problem (as long as $\mathscr{C}$ is polynomial-time evaluatable). Therefore, the characterization above implies that PAC learning is feasible in Algorithmica.

Corollary 2.6.3. *If $\mathsf{NP} \subseteq \mathsf{BPP}$, then every polynomial-time evaluatable concept class $\mathscr{C}$ is PAC learnable in polynomial time.*

NP Hardness of PAC Learning with Additional Requirements

Although finding a consistent hypothesis is sufficient for PAC learning, in Algorithmica, we can find a consistent hypothesis *with an additional condition* as long as its validity is verified in polynomial time. By Theorem 2.6.1, such an Occam algorithm yields a PAC learner with the same additional condition. Now, we present two previous results showing that the worst-case complexity of NP is characterized by PAC learning with such additional requirements on hypotheses.

Proper Learning

The first NP-hard learning problem is *proper* learning, where a learner for a concept class $\mathscr{C}$ must output a hypothesis contained in $\mathscr{C}$. Notice that finding a consistent hypothesis in the same class $\mathscr{C}$ is feasible if $\mathsf{NP} \subseteq \mathsf{BPP}$ and $\mathscr{C}$ is polynomial-time evaluatable. Pitt and Valiant [PV88] showed the NP-hardness of proper learning for a very simple class of 2-term DNF formulas (i.e., disjunctions of two monomials).

Theorem 2.6.4 ([PV88])**.** *The class of 2-term DNF formulas is properly PAC learnable in polynomial time if and only if* $\mathsf{NP} \subseteq \mathsf{BPP}$.

Furthermore, several followup studies presented NP-hardness results of proper learning for other classes and *semi*-proper learning (i.e., learning by a hypothesis class mildly larger than a concept class); e.g., proper learning for 3-node neural networks [BR93]; semi-proper learning for decision lists [HJLT96], DNF formulas [ABFKP08; Fel09], intersections of two halfspaces [KS11], and deterministic finite automatons [CLN14]; proper agonistic learning of halfspaces [GR09] and parities [GKS10].

In general, however, consideration of hypothesis classes larger enough than concept classes can change the difficulty of learning drastically. For instance, Pitt and Valiant [PV88] also showed that the class of 2-term DNF formulas is unconditionally PAC learnable by 2-CNF formulas. Particularly, showing NP-hardness of PAC learning in hypothesis-free setting is one of the central open problems in computational learning theory. For instance, Applebaum, Barak, and Xiao [ABX08] presented barrier results against standard proof frameworks for showing the NP-hardness of hypothesis-free PAC learning, such as Karp reductions and Turing reductions with constant adaptivity.

Learning by Minimum Programs

The second NP-hard learning problem is from the recent breakthrough result by Hirahara [Hir22a], who proved NP-hardness of MINLT that asks the minimum size of a program consistent with a given sample set. In fact, his result holds even in approximating the minimum size of the program that computes a parity and under P/poly-samplable distributions, which is stated as follows.

Theorem 2.6.5 ([Hir22a])**.** *The following promise problem is* NP*-complete (by randomized reductions): For given parameters* $1^n, 1^s$*, where* $n, s \in \mathbb{N}$*, and a circuit* $C\colon \{0,1\}^{\ell(n)} \to \{0,1\}^n \times \{0,1\}$*, let* $\mathcal{D}$ *be the distribution of* $C(r)$ *for* $r \sim \{0,1\}^{\ell(n)}$*, and determine the following:*

- (*Yes instances*) *There exists a program* M *of size* s *such that* M *computes a parity and*

$$\Pr_{(x,b)\sim\mathcal{D}}[M(x) = b] = 1.$$

- (*No instances*) *There exists no program* M *of size* $s \cdot n^{\epsilon}$ *such that*

$$\Pr_{(x,b)\sim\mathcal{D}}[M(x) = b] \leq 1/2 + 2^{-n^{1-\delta}},$$

where $\delta > 0$ *is an arbitrary constant and* $\epsilon = 1/(\log\log n)^{O(1)}$.

An immediate corollary to the result above is that $\mathsf{NP} \subseteq \mathsf{BPP}$ is derived from polynomial-time weak learning for parity functions under $\mathsf{P/poly}$-samplable distributions by the minimum consistent program (allowing the multiplicative approximation factor n^{ϵ}). One importance of this result lies in that the proof of Theorem 2.6.5 is provably *non-relativizing* (see Chapter 9 for further backgrounds of relativization). Therefore, even if we could exclude Heuristica and Pessiland by showing the learnability result above in Heuristica and Pessiland, it does not contradict the known relativization barriers against excluding Heuristica and Pessiland (see Section 9.3). This fact strongly motivates us to investigate the learning-theoretic aspects of Heuristica and Pessiland.

Chapter 3

Learning in Heuristica

In this chapter, we investigate how strong learning tasks can be feasible in *Heuristica*, i.e., learning-theoretic implications from the average-case easiness of NP. By definition, every learning task formulated directly as a distributional NP problem can be solvable in Heuristica. In Section 3.1, surprisingly, we demonstrate that even *worst-case* agnostic learning P/poly is feasible under the average-case errorless easiness of NP and a plausible computational assumption on example distributions. In Section 3.2, we present a new characterization of the average-case error-prone easiness of NP by a variant of the learning task called *extrapolation*.

3.1 Worst-Case Learning in Heuristica

In the original PAC learning model introduced by Valiant [Val84], a learner must satisfy two worst-case requirements: it must be *distribution-free* and must learn *every* target function. The worst-case nature of PAC learning becomes more apparent in the equivalent model of Occam learning [BEHW87; Sch90]. In Occam learning, a learner is given an arbitrary set of examples and is asked to find a small hypothesis consistent with all the given examples. Clearly, this task can be formulated as a (worst-case) search problem in NP. As seen in Chapter 2.6, the fundamental results of Blumer, Ehrenfeucht, Haussler, and Warmuth [BEHW87] and Schapire [Sch90] show that PAC learning and Occam learning are in fact equivalent.

Despite the worst-case nature of PAC learning and Occam learning, basing the hardness of these learning tasks on the hardness of NP has been a key challenges in computational learning theory for decades. The difficulty of proving the NP-hardness of learning has been explained in the work of Applebaum, Barak, and Xiao [ABX08], who showed that the NP-hardness of learning cannot be proved via a many-one reduction unless the polynomial hierarchy collapses. Given the lack of success in proving the NP-hardness of learning, it is natural to ask whether PAC learning is "NP-intermediate." In this section, we investigate whether PAC learning with *worst-case* requirements is feasible in Heuristica.

Previous studies in computational complexity has provided some insights into the relationship between learning and average-case complexity of NP. A line of studies [CIKK16; CIKK17; HS17; ILO20] on natural proofs and the Minimum Circuit Size Problem (MCSP) has revealed that learning with respect to the uniform distribution can be formulated as an average-case NP problem. Carmosino, Impagliazzo, Kabanets, and Kolokolova [CIKK16] presented a generic reduction from the task of PAC learning with respect to the uniform distribution to a natural property [RR97];

a natural property is essentially equivalent to solving MCSP on average [HS17] in the errorless setting. Therefore, these results imply that PAC learning for $\mathsf{P/poly}$ with respect to the uniform distribution is feasible under the assumption that MCSP $\in \mathsf{NP}$ is easy on average in the errorless setting.[1] Moreover, by combining this learning algorithm with inverters for distributional one-way functions [IL89], it can be shown that PAC learning with respect to every fixed samplable distribution on examples is feasible in (errorless) Heuristica (cf. [BCKRS22]). Nanashima [Nan21a] showed that, in (errorless) Heuristica, it is also possible to learn a target function chosen from a fixed samplable distribution with respect to every unknown example distribution (the result is found in Section 5.2). These results indicate that if either of the worst-case requirements on target functions or example distributions is weakened, then a polynomial-time learner for $\mathsf{P/poly}$ can be constructed from average-case errorless heuristics for NP problems.

Main Result in This Section

In this section, we present a PAC learner that satisfies the two worst-case requirements simultaneously in Heuristica. Under the assumption that NP admits average-case polynomial-time algorithms, we construct a polynomial-time learner that learns all polynomial-size circuits ($\mathsf{P/poly}$) with respect to all *unknown* efficiently samplable example distributions. In fact, our learning algorithm learns polynomial-size circuits *agnostically*, i.e., even if a target function is not in $\mathsf{P/poly}$, our learner outputs a hypothesis that is as good as the best hypothesis in $\mathsf{P/poly}$.

Theorem 3.1.1. *If* $\mathsf{DistNP} \subseteq \mathsf{AvgP}$, *then* $\mathsf{P/poly}$ *is agnostic learnable on all unknown* $\mathsf{P/poly}$*-samplable distributions in polynomial time.*

Let us remark on several points. First, our learner works without knowing example distributions; however, it needs to know an upper bound on the complexity of example distributions. Second, the running time of our learner depends on the complexity of the concept class and example distributions. Note that the complexity of a learner does not depend on an example distribution in the standard agnostic learning model. This is the only difference between the original learning model and our learning model of Theorem 3.1.1. Note that Theorem 3.1.1 is shown by relativizing techniques, i.e., the above-mentioned theorem holds in the presence of any oracle.

Relativization Barriers against Further Improvements

The result above naturally leads us to the question of whether the standard learner that does not depend on the complexity of example distributions can be constructed in Heuristica. Namely, can we remove the condition of Theorem 3.1.1 that example distributions must be $\mathsf{P/poly}$-samplable? In Section 9.1, we will present strong negative answers by constructing "relativized Heuristica" in which there is no PAC learner with respect to almost-uniform distributions. Here, we only present the statement.

Theorem 3.1.2. *For any arbitrary small constant* $\epsilon > 0$, *there exists an oracle* $\mathcal{O}_\epsilon$ *such that*

1. $\mathsf{DistPH}^{\mathcal{O}_\epsilon} \subseteq \mathsf{AvgP}^{\mathcal{O}_\epsilon}$;

[1] The learning algorithm of [CIKK16] requires a membership query. Ilango, Loff, and Oliveira [ILO20] showed that PAC learning with respect to the uniform distribution (without a membership query) is reduced to an average-case problem in NP.

2. $\mathsf{SIZE}^{\mathcal{O}_\epsilon}[n]$ *is not weakly learnable with membership queries on all uniform distributions over* $S \subseteq \{0,1\}^n$ *such that* $|S| > 2^{(1-\epsilon)n}$ *by nonuniform* $2^{o(n/\log n)}$*-time algorithms.*

This theorem shows that, unless we use some non-relativizing techniques, we cannot improve Theorem 3.1.1 for learning on almost-uniform example distributions even under the strong average-case assumption that $\mathsf{DistPH} \subseteq \mathsf{AvgP}$. Moreover, the hardness of learning holds even with the drastically weakened requirements: (a) weak learning (b) in sub-exponential time (c) with additional access to a membership query oracle and (d) an additional subexponential-length advice string (depending only on the example size).

Another question is whether we can obtain the same learnability result under the *error-prone* average-case easiness of NP (i.e., error-prone Heuristica). However, in Section 9.2, we will present strong negative answers by constructing "relativized *error-prone* Heuristica" in which there is no PAC learner with respect to the uniform distributions.

Theorem 3.1.3. *There exists an oracle* $\mathcal{O}$ *relative to which the following hold:*

1. $\mathsf{DistNP}^{\mathcal{O}} \subseteq \mathsf{HeurP}^{\mathcal{O}}$*;*
2. $\mathsf{SIZE}^{\mathcal{O}}[n]$ *is not weakly learnable with membership queries on the uniform example distribution by nonuniform* $2^{o(n/\log n)}$*-time algorithms;*
3. $\mathsf{P/poly}$ *is not weakly learnable with membership queries on the uniform example distribution by nonuniform* $2^n/n^{\omega(1)}$*-time algorithms.*

Note that $2^n/n^{O(1)}$ is the trivial upper bound on the running time for weak learning because a learner can identify a polynomial fraction of the truth-table of a target function by collecting $2^n/n^{O(1)}$ samples and weakly learn the target function by predicting examples outside of the identified truth-table uniformly at random. Namely, without the errorless condition on the average-case easiness of NP, we cannot beat the trivial algorithm for weak learning $\mathsf{P/poly}$ even on the uniform example distribution with additional query access to the membership query oracle.

In summary, both (i) the computational complexity assumption on the example distribution and (ii) the errorless condition are essential for relaxing the computational requirement for learning to average-case easiness in Theorem 3.1.1.

Overview of Proof Ideas

We explain the ideas for constructing the agnostic learner of Theorem 3.1.1 under the assumption that $\mathsf{DistNP} \subseteq \mathsf{AvgP}$. Our proofs are based on two results in previous studies.

The first result is the worst-case to average-case connection developed in [Hir18; Hir20b] for GapMINKT. We use the result in the form of Lemma 2.4.5.

Lemma (reminder of Lemma 2.4.5)**.** *If* $\mathsf{DistNP} \subseteq \mathsf{AvgP}$*, then there exist a polynomial* τ *and an algorithm* $\mathsf{ApproxK}_\tau$ *that is given* $(x, 1^t)$*, where* $x \in \{0,1\}^*, t \in \mathbb{N}$*, and outputs an integer* $s \in \mathbb{N}$ *in polynomial time to satisfy*

$$\mathrm{K}^{\tau(|x|+t)}(x) - \log \tau(|x|+t) \leq s \leq \mathrm{K}^t(x).$$

The second result is the characterization of agnostic learnability by random-right-hand-side refutation (RRHS-refutation) obtained by [KL18]. Remember that the task of correlative RRHS-refutation for $\mathscr{C}$ on a class $\mathscr{D}$ of example distributions is distinguishing the following two cases with

high probability: on input $\epsilon \in (0,1]$, (i) a "correlative" case where samples are chosen identically and independently according to $\mathsf{EX}_{f,\mathcal{D}_n}$ for $\mathcal{D}_n \in \mathscr{D}_n$ and a randomized function f such that $\mathrm{Cor}_{\mathcal{D}_n}(f,\mathscr{C}) \geq \epsilon$; and (ii) a "random" case where samples are chosen identically and independently according to $\mathsf{EX}_{f_R,\mathcal{D}_n}$ for $\mathcal{D}_n \in \mathscr{D}_n$ and a truly random function f_R. Kothari and Livni [KL18] showed that a concept class $\mathscr{C}$ is correlatively RRHS-refutable in polynomial time iff $\mathscr{C}$ is agnostic learnable in polynomial time (see Theorem 2.3.9).

In light of this characterization, our goal is to perform correlative RRHS-refutation using an approximation algorithm for time-bounded Kolmogorov complexity.

Now, we present a proof idea for constructing a correlative RRHS-refutation algorithm using an approximation algorithm $\mathsf{ApproxK}_\tau$ for time-bounded Kolmogorov complexity. Our refutation algorithm operates as follows:

1. For a given sample set $S = \{(x^{(i)}, b^{(i)})\}_{i=1}^m$, let $X = x^{(1)} \circ \cdots \circ x^{(m)}$ and $b = b^{(1)} \circ \cdots \circ b^{(m)}$.
2. Use $\mathsf{ApproxK}_\tau$ to approximate $\mathrm{K}^t(X)$ and $\mathrm{K}^{t'}(X \circ b)$ for some time bounds t and t', respectively. Let s and s' denote the respective approximated values.
3. If $\Delta = s' - s$ is less than some threshold T, then output "correlative"; otherwise, output "random".

We explain why this algorithm distinguishes the "correlative" case and the "random" case. In the former case, samples X and b are generated by a target function f such that $\mathrm{Cor}_D(f,\mathscr{C}) \geq \epsilon$. Thus, the best concept $c^* \in \mathscr{C}$ satisfies that $\Pr_{f,x\sim D}[c^*(x) \neq f(x)] \leq 1/2 - \epsilon/2$. Using this fact, we claim that the t'-time-bounded Kolmogorov complexity of $X \circ b$ is small for a properly large t'. Let $e \in \{0,1\}^m$ denote the string that indicates the difference between c^* and f, i.e., the i-th bit of e is $b^{(i)} \oplus c^*(x^{(i)})$ for every $i \in [m]$. Using the best concept c^*, a program d_X that describes X, and the string e that indicates an "error", we can describe the string $X \circ b$ by the following procedure: (1) compute X, (2) compute $b^* = c^*(x^{(1)}) \cdots c^*(x^{(m)})$ by applying c^* to each input $x^{(i)}$ contained in X, and (3) compute b (and output $X \circ b$) by taking bit-wise XOR between b^* and e. The length of the description of this procedure is bounded above by

$$|d_X| + |c^*| + |(\text{a description of } e)| + O(1) \leq s + \ell_{\mathscr{C}}(n) + (1 - \Omega(\epsilon^2)) \cdot m,$$

with high probability, where remember that $\ell_{\mathscr{C}}(n)$ is the length of the representation for $\mathscr{C}_n$. Therefore, it holds that $\Delta = s' - s$ is at most $\ell_{\mathscr{C}}(n) + (1 - \Omega(\epsilon^2)) \cdot m$ in a "correlative" case.

Thus, if $\Delta \approx m$ holds with high probability in a "random" case, then the algorithm distinguishes a "random" case from a "correlative" case by taking sufficiently large m with respect to $n, \ell_{\mathscr{C}}(n)$, and ϵ^{-1}. It seems reasonable to expect that $\Delta \approx m$ because b is a truly random string of m bits selected independently of X. However, in general, this might not hold for the following two technical reasons. First, we need nearly m bits to describe b with high probability; however, such b might help generate X in a time-bounded setting. Second, we must choose a time bound t' larger than t to ensure the upper bound on Δ in a "correlative" case, and this might also reduce the cost of generating X.

To analyze the case in which Δ becomes large, we consider the expected value of the *computational depth* of samples. Antunes, Fortnow, Melkebeek, and Vinodchandran [AFMV06] introduced the notion of the t-time-bounded computational depth of $x \in \{0,1\}^*$ (where $t \in \mathbb{N}$), which is defined as $\mathrm{K}^t(x) - \mathrm{K}(x)$. Hirahara [Hir21b] extended this notion to the (t,t')-time-bounded computational depth of $x \in \{0,1\}^*$ (where $t, t' \in \mathbb{N}$ with $t' > t$), which is defined as $\mathrm{K}^t(x) - \mathrm{K}^{t'}(x)$. Here, we further generalize these notions as follows:

Definition 3.1.4 (Sampling-depth functions)**.** *Let $t, t' \in \mathbb{N}$ such that $t' > t$. For a family of distributions $\mathcal{D}$, we define a (t,t')-sampling-depth function $sd^{t,t'}_{\mathcal{D}} = \{sd^{t,t'}_{\mathcal{D},n}\}_{n\in\mathbb{N}}$, where $sd^{t,t'}_{\mathcal{D},n} \colon \mathbb{N} \to \mathbb{R}_{\geq 0}$ by*

$$sd^{t,t'}_{\mathcal{D},n}(m) = \mathop{\mathbb{E}}_{X\sim\mathcal{D}^m_n}\left[\mathrm{K}^t(X) - \mathrm{K}^{t'}(X)\right].$$

We also extend the above-mentioned notion to a class of distributions. For a class $\mathscr{D}$ of families of distributions, we define a (t,t')-sampling-depth function $sd^{t,t'}_{\mathscr{D}} = \{sd^{t,t'}_{\mathscr{D},n}\}_{n\in\mathbb{N}}$, where $sd^{t,t'}_{\mathscr{D},n} \colon \mathbb{N} \to \mathbb{R}_{\geq 0}$ by

$$sd^{t,t'}_{\mathscr{D},n}(m) = \max_{\mathcal{D}\in\mathscr{D}} sd^{t,t'}_{\mathcal{D},n}(m).$$

We verify that if the sampling depth of example distributions is small, then Δ is large. We remark that Δ could become small because (i) the random string b and (ii) the larger time bound $t'(> t)$ could help generate X. However, if the sampling depth of the example distribution is small for t and t', the second case does not occur because $\mathrm{K}^{t'}(X)$ is close to $\mathrm{K}^t(X)$ with high probability. To show that the first case does not occur, we apply the weak symmetry of information (Theorem 2.4.2) that holds under the assumption that NP is easy on average. Informally speaking, by using the weak symmetry of information, we show that for any time bound $t \in \mathbb{N}$ and string X, and for a random string b, $\mathrm{K}^t(X \circ b)$ is larger than $\mathrm{K}^{t'}(X) + |b|$ for some large $t' > t$ with high probability over the choice of b. By the weak symmetry of information and the small sampling depth of the example distribution, we can show that $\mathrm{K}^{t'}(X \circ b)$ is large compared to $\mathrm{K}^t(X) + |b|$, i.e., the additional random string b does not help generate X so much.

To show Theorem 3.1.1, we will also observe that the sampling-depth function of a P/poly-samplable distribution is logarithmically small. Roughly speaking, this follows from the fact that samples selected according to a P/poly-samplable distribution have nearly optimal encoding with an efficient decoder, which can be proved using the techniques developed in [AF09; AGMMM18; Hir21b]. In other words, the term $\mathbb{E}[\mathrm{K}^t(X_{\mathcal{D}})]$ in the definition of a sampling-depth function is nearly close to $mH(\mathcal{D}) \approx \mathbb{E}[\mathrm{K}(X_{\mathcal{D}})]$ for a sufficiently large t.

Now, we formally prove Theorem 3.1.1 in subsequent two sections.

3.1.1 Agnostic Learning on Shallow Sampling-Depth Distributions

First, we construct the agnostic learner that works on distributions of shallow sampling-depth.

Theorem 3.1.5. *For any polynomial $\tau \colon \mathbb{N} \to \mathbb{N}$, there exist polynomials $p_\tau(n,m,t)$ and $p'_\tau(n,m,t)$ satisfying the following. If $\mathsf{DistNP} \subseteq \mathsf{AvgP}$, then there exists a learner L that agnostically learns $\mathscr{C}$ on $\mathscr{D}$ in time $\mathsf{poly}(n, m(n,\epsilon/2), t(n,\epsilon/2), \epsilon^{-1})$ with sample complexity $O(\epsilon^{-2}\cdot m(n,\epsilon/2)^3)$, where $m, t \colon \mathbb{N}\times(0,1] \to \mathbb{N}$ are arbitrary functions satisfying the following conditions: for all sufficiently large n and for all $\epsilon \in (0,1]$,*

$$t(n,\epsilon) \geq p_0(nm(n,\epsilon)^2), \text{ and}$$
$$m(n,\epsilon) > \frac{8}{\epsilon^2}\left(n + \ell_{\mathscr{C}}(n) + 6sd^{t(n,\epsilon),p_\tau(n,m(n,\epsilon),t(n,\epsilon))}_{\mathscr{D},n}(m(n,\epsilon)) + \log p'_\tau(n,m(n,\epsilon),t(n,\epsilon))\right).$$

Proof. Let $m := m(n,\epsilon)$ and $t := t(n,\epsilon)$. First, we specify the polynomials p_τ and p'_τ. Fix $x^{(1)},\dots,x^{(m)} \in \{0,1\}^n$ and $f \in \mathscr{C}_n$ arbitrarily. Let $X = x^{(1)} \circ \cdots \circ x^{(m)}$. Then, we can compute $f(x^{(1)}),\dots,f(x^{(m)})$ in time $m\cdot\mathsf{poly}(n)$ from X, the representation of f, and the evaluation algorithm for $\mathscr{C}$ (where we use the assumption that $\ell_{\mathscr{C}}(n) \leq \mathsf{poly}(n)$ and $\mathscr{C}$ is polynomially evaluatable). For

any $b \in \{0,1\}^m$ such that $|\{i \in [m] : b_i = f(x^{(i)})\}| \geq (1/2 + \epsilon/4)m$, we define $e \in \{0,1\}^m$ by $e_i = b_i \oplus f(x^{(i)})$. Then, e is reconstructed from $H_2(1/2 + \epsilon/4)m$ bits in time $\mathsf{poly}(n, m)$ by lexicographic indexing among binary strings of the same weight, where H_2 is the binary entropy function. Therefore, we can take a polynomial $t'(n, m, t)$ such that, for any sufficiently large n,

$$\begin{aligned} \mathrm{K}^{t'(n,m,t)}(X \circ b) &\leq \mathrm{K}^{\tau(nm+t)}(X) + \ell_{\mathscr{C}}(n) + n + H_2(1/2 + \epsilon/4)m \\ &\leq \mathrm{K}^{\tau(nm+t)}(X) + \ell_{\mathscr{C}}(n) + n + (1 - \epsilon^2/8) \cdot m, \end{aligned} \tag{3.1}$$

where we applied the Taylor series of H_2 in a neighborhood of $1/2$, i.e., for any $\delta \in [-1/2, 1/2]$,

$$H_2(1/2 + \delta) = 1 - \frac{1}{2 \ln 2} \sum_{i=1}^{\infty} \frac{(2\delta)^{2i}}{i(2i-1)} \leq 1 - \frac{2}{\ln 2}\delta^2 \leq 1 - 2\delta^2.$$

Now, we define the polynomials p_τ and p'_τ by

$$\begin{aligned} p_\tau(n, m, t) &= p_w(6\tau(nm + t'(n, m, t))), \text{ and} \\ p'_\tau(n, m, t) &= p_w(6\tau(nm + t'(n, m, t)))\tau(nm + t'(n, m, t))\tau(nm + t) \end{aligned}$$

Next, we construct a refutation algorithm R for $\mathscr{C}$ as follows. On input $1^n, 1^{\epsilon^{-1}}$ (where $n, \epsilon^{-1} \in \mathbb{N}$) and a sample set $S = \big((x^{(1)}, b^{(1)}), \ldots, (x^{(m)}, b^{(m)})\big)$, the refuter R computes t and $t' := t'(n, m, t)$, executes $s \leftarrow \mathsf{ApproxK}_\tau(X, 1^t)$ and $s' \leftarrow \mathsf{ApproxK}_\tau(X \circ b, 1^{t'})$ for $X = x^{(1)} \circ \cdots \circ x^{(m)}$ and $b = b^{(1)} \circ \cdots \circ b^{(m)}$, and finally outputs "correlative" if $s' - s \leq m + \ell_{\mathscr{C}}(n) + n + \log \tau(nm + t) - m\epsilon^2/8$ and outputs "random" otherwise.

We can easily verify that R halts in polynomial time in n, m, and t. We now verify the correctness of R. Let f denote a target randomized function for refutation.

In "correlative" cases, there exists a function $f^* \in \mathscr{C}_n$ such that

$$\Pr_{x \sim \mathcal{D}, f}[f(x) = f^*(x)] = \frac{1}{2} + \frac{\mathrm{Cor}_{\mathcal{D}}(f, \mathscr{C})}{2} \geq \frac{1}{2} + \frac{\epsilon}{2}.$$

According to the Hoeffding inequality, the probability that $|\{i \in [m] : b^{(i)} = f^*(x^{(i)})\}| < 1/2 + \epsilon/4$ holds is less than $\exp(-2m \cdot (\epsilon/4)^2) \leq \exp(-n \cdot 8/\epsilon^2 \cdot \epsilon^2/8) \leq 1/3$ over the choice of S for any sufficiently large $n \in \mathbb{N}$. In such cases, by Lemma 2.4.5 and Eq. (3.1), we have

$$\begin{aligned} s' &\leq \mathrm{K}^{t'(n,m,t)}(X \circ b) \\ &\leq \mathrm{K}^{\tau(nm+t)}(X) + \ell_{\mathscr{C}}(n) + n + (1 - \epsilon^2/8) \cdot m \\ &\leq s + \log \tau(nm + t) + \ell_{\mathscr{C}}(n) + n + (1 - \epsilon^2/8) \cdot m, \end{aligned}$$

and

$$s' - s \leq m + \ell_{\mathscr{C}}(n) + n + \log \tau(nm + t) - m\epsilon^2/8.$$

Thus, $R(1^n, 1^{\epsilon^{-1}}, S)$ outputs "correlative" with a probability of at least $2/3$.

In "random" cases, b is selected uniformly at random from $\{0,1\}^m$. By the assumption that $\mathsf{DistNP} \subseteq \mathsf{AvgP}$, $t \geq p_0(nm \cdot m)$, and the weak symmetry of information (Theorem 2.4.2), for any $X \in \{0,1\}^{nm}$, we have

$$\Pr_b\left[\mathrm{K}^{\tau(nm+t'(n,m,t))}(X \circ b) \geq \mathrm{K}^{p_w(6\tau(nm+t'(n,m,t)))}(X) + m - \log p_w(6\tau(nm + t'(n, m, t)))\right] \leq 1/6.$$

Let $\mathcal{D} \in \mathscr{D}$ be an arbitrary example distribution. By Markov's inequality, we can show that

$$\Pr_X\left[\mathrm{K}^t(X) - \mathrm{K}^{p_\tau(n,m,t)}(X) > 6sd^{t,p_\tau(n,m,t)}_{\mathscr{D},n}(m)\right] \leq \frac{sd^{t,p_\tau(n,m,t)}_{\mathcal{D},n}(m)}{6sd^{t,p_\tau(n,m,t)}_{\mathscr{D},n}(m)} \leq \frac{1}{6}.$$

Thus, the following inequality holds with a probability of at least $1 - (1/6 + 1/6) = 2/3$:

$$\begin{aligned}
s' &\geq \mathrm{K}^{\tau(nm+t'(n,m,t))}(X \circ b) - \log \tau(nm + t'(n,m,t)) \\
&\geq \mathrm{K}^{p_w(6\tau(nm+t'(n,m,t)))}(X) + m - \log p_w(6\tau(nm + t'(n,m,t)))\tau(nm + t'(n,m,t)) \\
&\geq \mathrm{K}^{p_\tau(n,m,t)}(X) + m - \log p'_\tau(n,m,t) + \log \tau(nm + t) \\
&\geq \mathrm{K}^t(X) - (\mathrm{K}^t(X) - \mathrm{K}^{p_\tau(n,m,t)}(X)) + m - \log p'_\tau(n,m,t) + \log \tau(nm + t) \\
&\geq s - (\mathrm{K}^t(X) - \mathrm{K}^{p_\tau(n,m,t)}(X)) + m - \log p'_\tau(n,m,t) + \log \tau(nm + t) \\
&\geq s - 6sd^{t,p_\tau(n,m,t)}_{\mathscr{D},n}(m) + m - \log p'_\tau(n,m,t) + \log \tau(nm + t).
\end{aligned}$$

By arranging the above, we get

$$s' - s \geq m - 6sd^{t,p_\tau(n,m,t)}_{\mathscr{D},n}(m) - \log p'_\tau(n,m,t) + \log \tau(nm + t).$$

By the assumption that

$$m > \frac{8}{\epsilon^2}\left(n + \ell_{\mathscr{C}}(n) + 6sd^{t,p_\tau(n,m,t)}_{\mathscr{D},n}(m) + \log p'_\tau(n,m,t)\right),$$

we have

$$\begin{aligned}
&\left(m - 6sd^{t,p_\tau(n,m,t)}_{\mathscr{D},n}(m) - \log p'_\tau(n,m,t) + \log \tau(nm + t)\right) - \left(m + \ell_{\mathscr{C}}(n) + n + \log \tau(nm + t) - m\epsilon^2/8\right) \\
&= m\epsilon^2/8 - \left(n + \ell_{\mathscr{C}}(n) + 6sd^{t,p_\tau(n,m,t)}_{\mathscr{D},n}(m) + \log p'_\tau(n,m,t)\right) \\
&> 0.
\end{aligned}$$

Thus, $s' - s > m + \ell_{\mathscr{C}}(n) + n + \log \tau(nm + t) - m\epsilon^2/8$ holds, and R outputs "random" in such cases. Therefore, $R(1^n, 1^{\epsilon^{-1}}, S)$ outputs "random" with a probability of at least $2/3$

By the above-mentioned argument, R correlatively RRHS-refutes $\mathscr{C}$ on $\mathscr{D}$ in time $\mathsf{poly}(n,m,t)$ with m samples. Thus, by Theorem 2.3.9, we conclude that $\mathscr{C}$ is agnostic learnable on $\mathscr{D}$ in time

$$O\left(\mathsf{poly}(n, m(n,\epsilon/2), t(n,\epsilon/2)) \cdot \frac{m(n,\epsilon/2)^2}{\epsilon^2}\right) = \mathsf{poly}(n, m(n,\epsilon/2), t(n,\epsilon/2), \epsilon^{-1}),$$

with $O(\frac{m(n,\epsilon/2)^3}{\epsilon^2})$ samples. □

3.1.2 Sampling-Depth of P/poly-Samplable Distributions

Next, we observe that P/poly-samplable distributions have a logarithmically small sampling depth. Then, we use Theorem 3.1.5 to establish the agnostic learnability on P/poly-samplable distributions.

To analyze the sampling depth of $\mathsf{Samp}[t(n)]/a(n)$, we introduce the following useful lemmas.

Lemma 3.1.6 ([AGMMM18; Hir20a, Corollary 9.8])**.** *If* $\mathsf{DistNP} \subseteq \mathsf{AvgP}$*, then there exists a polynomial* $p\colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ *such that for any* $t, a\colon \mathbb{N} \to \mathbb{N}$*,* $n, m \in \mathbb{N}$*,* $\mathcal{D}_n \in \mathsf{Samp}[t(n)]/a(n)_n$*, and* $x \in \mathrm{supp}(\mathcal{D}_n^m)$*,*

$$K^{p(t(n),m)}(x) \leq -\log \mathcal{D}_n^m(x) + O(\log m) + O(\log t(n)) + a(n).$$

The following holds by the noiseless coding theorem.

Lemma 3.1.7 (cf. [LV19, Theorem 8.1.1])**.** *For any distribution* $\mathcal{D}$*,* $\mathbb{E}_{x\sim\mathcal{D}}[\mathrm{K}(x)] \geq H(\mathcal{D})$*.*

Now, we show the upper bound on the sampling depth of $\mathsf{Samp}[t(n)]/a(n)$.

Lemma 3.1.8. *If* $\mathsf{DistNP} \subseteq \mathsf{AvgP}$*, then there exists a polynomial* $p_0'\colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ *such that for any* $t, a\colon \mathbb{N} \to \mathbb{N}$*,* $n, m \in \mathbb{N}$*, the following expression holds: for all* $t' \geq p_0'(t(n), m)$*,*

$$sd^{t',\infty}_{\mathsf{Samp}[t]/a,n}(m) \leq O(\log m + \log t(n)) + a(n).$$

In this section, we use the notation p_0' to refer to the polynomial in Lemma 3.1.8.

Proof. Fix $\mathcal{D} \in \mathsf{Samp}[t(n)]/a(n)$ arbitrarily. Let p_0' denote the polynomial p in Lemma 3.1.6. Assuming that $\mathsf{DistNP} \subseteq \mathsf{AvgP}$ and Lemma 3.1.6, for each $m \in \mathbb{N}$, we have

$$\begin{aligned}
\mathop{\mathbb{E}}_{x\sim\mathcal{D}_n^m}[\mathrm{K}^{t'}(x)] &\leq \mathop{\mathbb{E}}_{x\sim\mathcal{D}_n^m}[\mathrm{K}^{p_0'(t(n),m)}(x)] \\
&\leq \mathop{\mathbb{E}}_{x\sim\mathcal{D}_n^m}[-\log \mathcal{D}_n^m(x)] + O(\log m + \log t(n)) + a(n) \\
&\leq H(\mathcal{D}_n^m) + O(\log m + \log t(n)) + a(n) \\
&\leq \mathop{\mathbb{E}}_{x\sim\mathcal{D}_n^m}[\mathrm{K}(x)] + O(\log m + \log t(n)) + a(n).
\end{aligned}$$

Thus, we conclude that

$$\begin{aligned}
sd^{t',\infty}_{\mathcal{D},n}(m) &= \mathop{\mathbb{E}}_{x\sim\mathcal{D}_n^m}[\mathrm{K}^{t'}(x)] - \mathop{\mathbb{E}}_{x\sim\mathcal{D}_n^m}[\mathrm{K}(x)] \\
&\leq O(\log m + \log t(n)) + a(n).
\end{aligned}$$

□

Theorem 3.1.5 and Lemma 3.1.8 imply the following corollary, which is the formal statement of Theorem 3.1.1.

Corollary 3.1.9. *If* $\mathsf{DistNP} \subseteq \mathsf{AvgP}$*, then for any polynomials* $s, t_s, a_s\colon \mathbb{N} \to \mathbb{N}$*,* $\mathsf{SIZE}[s(n)]$ *is agnostic learnable on* $\mathsf{Samp}[t_s(n)]/a_s(n)$ *in polynomial time with sample complexity* $O(\epsilon^{-8+\Delta}(n + s(n) + a_s(n))^{3+\Delta})$*, where* $\Delta > 0$ *is an arbitrary small constant.*

We remark that the time complexity t_s for the sampling algorithms above affects only the time complexity of the agnostic learner.

Proof. Let $\Delta > 0$ be an arbitrary small constant. By the assumption that $\mathsf{DistNP} \subseteq \mathsf{AvgP}$ and Theorem 2.4.4, there exists a polynomial τ such that $\mathrm{Gap}_\tau\mathrm{MINKT} \in \text{pr-}\mathsf{P}$. Thus, we can apply Theorem 3.1.5.

We define the functions $m, t\colon \mathbb{N} \times (0,1] \to \mathbb{N}$ by

$$\begin{aligned} m(n,\epsilon) &= (\epsilon^{-2}\log\epsilon^{-1} \cdot (n + s(n)\log s(n) + a_s(n)))^{1+\Delta}, \text{ and} \\ t(n,\epsilon) &= \max\{\lceil p_0(n \cdot m(n,\epsilon)^2)\rceil, \lceil p_0'(t_s(n), m(n,\epsilon))\rceil\}. \end{aligned}$$

Obviously, $t(n,\epsilon) > p_0(n \cdot m(n,\epsilon)^2)$ and $t(n,\epsilon) > p_0'(t_s(n), m(n,\epsilon))$ hold. By Lemma 3.1.8, we get

$$\begin{aligned} sd^{t(n,\epsilon),p_\tau(n,m(n,\epsilon),t(n,\epsilon))}_{\mathsf{Samp}[t_s]/a_s,n}(m(n,\epsilon)) &\le sd^{t(n,\epsilon),\infty}_{\mathsf{Samp}[t_s]/a_s,n}(m(n,\epsilon)) \\ &\le O(\log m(n,\epsilon) + \log t_s(n)) + a_s(n). \end{aligned}$$

It is easily verified that $t(n,\epsilon) \le \mathsf{poly}(n, s(n), t_s(n), a_s(n), \epsilon^{-1}) = \mathsf{poly}(n, \epsilon^{-1})$ and

$$\log p_\tau'(n, m(n,\epsilon), t(n,\epsilon)) \le O(\log n + \log\epsilon^{-1} + \log m(n,\epsilon)).$$

Therefore, for any sufficiently large $n \in \mathbb{N}$, we have

$$\begin{aligned} &\frac{8}{\epsilon^2}\left(n + O(s(n)\log s(n)) + 6sd^{t(n,\epsilon),p_\tau(n,m(n,\epsilon),t(n,\epsilon))}_{\mathsf{Samp}[t_s]/a_s,n}(m(n,\epsilon)) + \log p_\tau'(n, m(n,\epsilon), t(n,\epsilon))\right) \\ &\le \epsilon^{-2} \cdot O(n + s(n)\log s(n) + \log\epsilon^{-1} + a_s(n) + \log m(n,\epsilon)) \\ &\le O(\epsilon^{-2}\log\epsilon^{-1}(n + s(n)\log s(n) + a_s(n))) \\ &\le m(n,\epsilon). \end{aligned}$$

Thus, by Theorem 3.1.5, we conclude that $\mathsf{SIZE}[s(n)]$ is agnostic learnable in time

$$\mathsf{poly}(n, m(n,\epsilon/2), t(n,\epsilon/2), \epsilon^{-1}) = \mathsf{poly}(n, s(n), \epsilon^{-1}, t_s(n), a_s(n)) = \mathsf{poly}(n, \epsilon^{-1}).$$

The sample complexity is at most $O(\epsilon^{-2}m(n,\epsilon/2)^3)$, which is bounded above by $O(\epsilon^{-8+\Delta'}(n+s(n)+a_s(n))^{3+\Delta'})$ for an arbitrary small constant $\Delta' > 0$ by selecting a sufficiently small Δ compared to Δ' in the above-mentioned argument. □

Remark 3.1.10. In fact, it is not clear whether several techniques (e.g., the weak symmetry of information) developed in [Hir20b; Hir21b] can be relativized when we only assume that $\mathsf{DistNP} \subseteq \mathsf{AvgP}$, owing to the pseudorandom generator construction presented in [BFP05]. However, all of them can be relativized under the stronger assumption that $\mathsf{Dist}\Sigma_2^p \subseteq \mathsf{AvgP}$ (refer to [HN21, Appendix]). Thus, Theorem 3.1.5 and all the results in this section can be also relativized under the assumption that $\mathsf{Dist}\Sigma_2^p \subseteq \mathsf{AvgP}$. Furthermore, when we restrict the target to the efficient agnostic learning on P/poly-samplable distributions (i.e., Theorem 3.1.1), we can obtain the same learnability result by using only relativized techniques from the following observations. First, the same upper bound of the sampling-depth function in Lemma 3.1.8 is obtained by applying the encoding developed in [AF09; AGMMM18] with additional random strings, and such additional random strings are available for learners. Second, the same upper bound of the sampling-depth function in Lemma 3.1.8 holds for $t' = \infty$; in this case, we can apply the symmetry of information for resource-unbounded Kolmogorov complexity instead of the weak symmetry of information in Theorem 3.1.5. Third, since the upper bound in Lemma 3.1.8 is logarithmically small, the algorithm for GapMINKT in [Hir18] with a worse approximation factor is sufficient, which can be relativized.

3.2 Conditional Extrapolation

In this section, we study learning-theoretic consequences of the average-case *error-prone* easiness of NP, where we introduce another natural learning task named conditional extrapolation, inspired by the notion *extrapolation* introduced by Impagliazzo and Levin [IL90]. Further background of the previous work [IL90] can be found in Chapter 4.

Informally, we define conditional extrapolation as a task of, for a given conditional string x, to statistically simulate a string subsequent to x under some distribution $\mathcal{D}$. For instance, when $\mathcal{D}$ is the distribution of music composed by Mozart, by conditional extrapolation, we can statistically simulate how Mozart continues a given first eight measures x of, e.g., a song by Billie Eilish or Japanese traditional song *enka*. Here, we consider the average-case setting over the conditional string x, where x is selected by another samplable distribution $\mathcal{C}$. Then, we can show that the feasibility of conditional extrapolation *exactly* characterizes the average-case error-prone easiness of NP!

Theorem 3.2.1. *The following are equivalent.*

1. $\mathsf{DistNP} \subseteq \mathsf{HeurBPP}$.
2. **(Conditional Extrapolation)** *For every samplable distribution families $\{\mathcal{C}_n\}_{n\in\mathbb{N}}$ and $\{\mathcal{D}_n\}_{n\in\mathbb{N}}$ over binary strings, there exists a polynomial-time randomized algorithm* Ext *such that for all $n, k, \epsilon^{-1}, \delta^{-1} \in \mathbb{N}$,*

$$\Pr_{x\sim\mathcal{C}_n}\left[\mathsf{L}_1\left(\mathsf{Ext}(x; 1^{\langle k,\epsilon^{-1},\delta^{-1}\rangle}), \mathsf{Next}_k(x, \mathcal{D}_n)\right) \leq \epsilon\right] \geq 1-\delta.$$

Note that Impagliazzo and Levin [IL90] characterized the existence of OWF in the special case in which $\mathcal{C}_n$ corresponds to the prefix of $\mathcal{D}_n$. In this sense, the result above generalizes their result. In fact, the proof mainly follows from the well-known equivalence result between the existence of OWF and the existence of distributional OWF [IL89] except that we use a heuristic scheme for the circuit SAT problem (i.e., the average-case easiness of NP) instead of inverting algorithms.

3.2.1 From Conditional Extrapolation to Average-Case Easiness of NP

We first show a direction from conditional extrapolation to the average-case easiness of NP.

Lemma 3.2.2 (Item 2 $\Rightarrow$ Item 1 in Theorem 3.2.1)**.** $\mathsf{DistNP} \subseteq \mathsf{HeurBPP}$ *holds if for every samplable distribution families $\{\mathcal{C}_n\}_{n\in\mathbb{N}}$ and $\{\mathcal{D}_n\}_{n\in\mathbb{N}}$ over binary strings, there exists a polynomial-time randomized algorithm* Ext *such that for all $n, k, \epsilon^{-1}, \delta^{-1} \in \mathbb{N}$,*

$$\Pr_{x\sim\mathcal{C}_n}\left[\mathsf{L}_1\left(\mathsf{Ext}(x; 1^{\langle k,\epsilon^{-1},\delta^{-1}\rangle}), \mathsf{Next}_k(x, \mathcal{D}_n)\right) \leq \epsilon\right] \geq 1-\delta.$$

Proof. Let $(L, \mathcal{C})$ be an arbitrary distributional NP problem, where $\mathcal{C} = \{\mathcal{C}_n\}_{n\in\mathbb{N}} \in \mathrm{PSAMP}$. Let V_L be a verifier for L. Without loss of generality, we assume that there exists a polynomial p such that for each $n \in \mathbb{N}$ and each $x \in \operatorname{supp} \mathcal{D}_n$, there exists $y \in \{0,1\}^{p(n)}$ satisfying that $V(x, y) = 1$ if and only if $x \in L$ (by a simple padding argument).

We define another samplable distribution $\mathcal{D} = \{\mathcal{D}_n\}_{n\in\mathbb{N}}$ as follows: each $\mathcal{D}_n$ is a distribution of $x \circ V_L(x,y) \circ y$, where $x \sim \mathcal{C}_n$ and $y \sim \{0,1\}^{p(n)}$. By the assumption of conditional extrapolation, there exists a randomized algorithm Ext such that for all $n, \epsilon^{-1}, \delta^{-1} \in \mathbb{N}$

$$\Pr_{x\sim\mathcal{C}_n}\left[\mathsf{L}_1\left(\mathsf{Ext}(x1; 1^{\langle p(n),\epsilon^{-1},\delta^{-1}\rangle}), \mathsf{Next}_{p(n)}(x1, \mathcal{D}_n)\right) \le \epsilon\right] \ge 1-\delta.$$

Notice that for every $x \in \operatorname{supp}\mathcal{C}_n$, if $x \in L$, then $\mathsf{Next}_{p(n)}(x1, \mathcal{D}_n)$ is the uniform distribution over witness strings y such that $V_L(x,y) = 1$; otherwise if $x \notin L$, then $\mathsf{Next}_{p(n)}(x1, \mathcal{D}_n)$ is the distribution over $\{\varepsilon\}$.

We define a randomized heuristic scheme A for $(L, \mathcal{C})$ as follows: On input $x \in \operatorname{supp}\mathcal{C}_n$ and parameters $n, \delta^{-1} \in \mathbb{N}$, the algorithm A samples $y \sim \mathsf{Ext}(x; 1^{\langle p(n),1/4,\delta^{-1}\rangle})$ and outputs $V_L(x,y)$. It is easy to verify that A halts in polynomial time in n and δ^{-1}.

For correctness, it suffice to show that $A(x; 1^n, 1^{\delta^{-1}})$ outputs $L(x)$ with probability at least $3/4$ (over the choice of randomness for Ext) under the condition that

$$\mathsf{L}_1\left(\mathsf{Ext}(x; 1^{\langle p(n),1/4,\delta^{-1}\rangle}), \mathsf{Next}_{p(n)}(x, \mathcal{D}_n)\right) \le 1/4$$

because the conditional event occurs at least $1-\delta$ over the choice of $x \sim \mathcal{C}_n$. If $x \notin L$, then A always output 0 because no y satisfies $V_L(x,y) = 1$. If $x \in L$, then

$$\begin{aligned}\Pr_A[A(x; 1^n, 1^{\delta^{-1}}) = 1] &= \Pr_{y\sim\mathsf{Ext}(x;1^{\langle p(n),1/4,\delta^{-1}\rangle})}[V_L(x,y) = 1] \\ &\ge \Pr_{y\sim\mathsf{Next}_{p(n)}(x,\mathcal{D}_n)}[V_L(x,y) = 1] - 1/4 \\ &= 1 - 1/4 = 3/4.\end{aligned}$$

Thus, we conclude that $(L, \mathcal{C}) \in \mathsf{HeurBPP}$. □

3.2.2 From Average-Case Easiness of NP to Conditional Extrapolation

Next, we show the opposite direction, i.e., from the average-case easiness of NP to conditional extrapolation by generalizing the proof by Impagliazzo and Rudich [IR89].

Lemma 3.2.3 (Item 1 ⇒ Item 2 in Theorem 3.2.1)**.** *If* $\mathsf{DistNP} \subseteq \mathsf{HeurBPP}$ *holds, then for every samplable distribution families* $\{\mathcal{C}_n\}_{n\in\mathbb{N}}$ *and* $\{\mathcal{D}_n\}_{n\in\mathbb{N}}$ *over binary strings, there exists a polynomial-time randomized algorithm* Ext *such that for all* $n, k, \epsilon^{-1}, \delta^{-1} \in \mathbb{N}$*,*

$$\Pr_{x\sim\mathcal{C}_n}\left[\mathsf{L}_1\left(\mathsf{Ext}(x; 1^{\langle k,\epsilon^{-1},\delta^{-1}\rangle}), \mathsf{Next}_k(x, \mathcal{D}_n)\right) \le \epsilon\right] \ge 1-\delta.$$

To show Lemma 3.2.3, we use the following decision-to-search reduction.

Theorem 3.2.4 (Search-to-decision reduction [BCGL92])**.** *If* $\mathsf{DistNP} \subseteq \mathsf{HeurBPP}$*, then for every* $(L, \mathcal{D}) \in \mathsf{DistNP}$*, where* L *is determined by an* NP *relation* $R_L \subseteq \{0,1\}^* \times \{0,1\}^*$*, there exists a polynomial-time randomized algorithm* M *such that for every* $n, \epsilon^{-1} \in \mathbb{N}$*,*

$$\Pr_{x\sim\mathcal{D}_n}\left[x \notin L \vee \Pr_M[R_L(x, M(x, 1^n, 1^{\epsilon^{-1}}))] \ge 1 - 2^{-n}\right] \ge 1-\epsilon.$$

Proof of Lemma 3.2.3. Let $\mathcal{C} = \{\mathcal{C}_n\}_{n\in\mathbb{N}}$ and $\mathcal{D} = \{\mathcal{D}_n\}_{n\in\mathbb{N}}$ be arbitrary samplable distribution families. Let $\ell(n)$ be a polynomial that represents the seed length required for sampling according to $\mathcal{D}_n$. Based on the standard way to transform Turing machines into uniformly computable circuits, we obtain a uniformly computable polynomial-size circuit family $D = \{D_n\}_{n\in\mathbb{N}}$, where $D_n\colon \{0,1\}^{\ell(n)} \to \{0,1\}^{t(n)}$ for each $n \in \mathbb{N}$ such that the distribution of $D_n(r)$ for $r \sim \{0,1\}^{\ell(n)}$ is statistically equivalent to $\mathcal{D}_n$, where $t(n)$ is a polynomial, and we fix the output length of D_n by a simple padding argument.

We define an NP language $\mathsf{InvCirc}$ as follows: For every $x \in \{0,1\}^*$ and every (binary representation of) circuit C,

$$\langle x, C\rangle \in \mathsf{InvCirc} \iff \exists r \in \{0,1\}^* \text{ such that } C(r) = x.$$

We define a samplable distribution $\mathcal{E} = \{\mathcal{E}_n\}_{n\in\mathbb{N}}$ as follows. For each $n, m \in \mathbb{N}$, let $\mathcal{H}_{n,m}$ be the pairwise independent hash family that maps n bits to m bits. For each $n \in \mathbb{N}$, we define $\mathcal{E}_n$ as a distribution of $\langle\langle h, i, x, v\rangle, E_n\rangle$, where $h \sim \mathcal{H}_{\ell(n),\ell(n)+\log^2 n}$, $i \sim [\ell(n) + \log^2 n]$, $x \sim \mathcal{C}_n$, $v \sim \{0,1\}^i$, and E_n denotes a circuit that is given $r \in \{0,1\}^{\ell(n)}, h' \in \mathcal{H}_{\ell(n),\ell(n)+\log^2 n}, i' \in [\ell(n) + \log^2 n]$ and outputs $\langle h', i', D_n(r)_{[|x|]}, h'(r)_{[i']}\rangle$. Here, we assume that $\langle h, i, x, v\rangle$ and $\langle h', i', D_n(r)_{[|x|]}, h'(r)_{[i']}\rangle$ are encoded to the same polynomial length for each n by the zero-padding. Since D_n is uniformly computable in polynomial time, the distribution family $\mathcal{E}$ is samplable, and $(\mathsf{InvCirc}, \mathcal{E}) \in \mathsf{DistNP}$.

By the assumption that $\mathsf{DistNP} \subseteq \mathsf{HeurBPP}$ and Theorem 3.2.4, there exists a polynomial-time randomized algorithm M such that for every $n, \epsilon^{-1} \in \mathbb{N}$,

$$\Pr_{z=\langle\langle h,i,y,v\rangle,E_n\rangle\sim\mathcal{E}_n}\left[\langle\langle h, i, y, v\rangle, E_n\rangle \notin \mathsf{InvCirc} \vee \Pr_M[E_n(M(z, 1^n, 1^{\epsilon^{-1}})) = \langle h, i, y, v\rangle] \geq 1 - 2^{-n}\right] \geq 1-\epsilon.$$

For simplicity, we assume that M does not fail, i.e., we assume that

$$\Pr_{z=\langle\langle h,i,y,v\rangle,E_n\rangle\sim\mathcal{E}_n}\left[\langle h, i, y, v\rangle \notin \mathrm{Im}E_n \vee E_n(M(z, 1^n, 1^{\epsilon^{-1}})) = \langle h, i, y, v\rangle\right] \geq 1 - \epsilon.$$

This additional assumption is valid in the following sense: The algorithm Ext (specified later) executes M only polynomially many times. Thus, the probability that M does not satisfy the above is negligible. Therefore, this affects the result negligibly, and we can manage this by selecting slightly better parameters.

The outline of the proof is the following: First, we construct an algorithm M' that approximates the number of consistent inverses of a given $y \sim \mathcal{C}_n$ with respect to D_n based on M. Then, we construct the extrapolation algorithm Ext based on M and M'.

We construct the approximation algorithm M' as follows: On input $x \sim \mathcal{C}_n$ and $1^{\delta^{-1}}$, where $\delta^{-1} \in \mathbb{N}$ is an additional error parameter, M' executes $M(\langle\langle h_{i,j}, i, x, v_{i,j}\rangle, E_n\rangle, 1^n, 1^{(\ell(n)+\log^2 n)\gamma^2})$ for each $i \in [\ell(n) + \log^2 n]$ and $j \in [\ell(n)]$, where $h_{i,j} \sim \mathcal{H}_{\ell(n),\ell(n)+\log^2 n}$, $v_{i,j} \sim \{0,1\}^i$, and $\gamma := \max\{\delta^{-1}, 16\}$. Then, M' outputs the maximum value of i (denoted by i^*) satisfying that M succeeds in finding an inverse with respect to E_n for some $j \in [\ell(n)]$, i.e., i^* is the maximum value of i satisfying that there exists $j \in [\ell(n)]$ such that

$$E_n(M(\langle h_{i,j}, i, x, v_{i,j}\rangle, 1^n, 1^{(\ell(n)+\log^2 n)\gamma^2})) = \langle h_{i,j}, i, x, v_{i,j}\rangle.$$

If there is no such i, then let $i^* = 0$.

It is easy to verify that M' halts in polynomial time. Let $R_x = \{r \in \{0,1\}^{\ell(n)} : D(r)_{[|x|]} = x\}$, $N_x := |R_x|$, and $n_x := \lfloor \log N_x \rfloor$. Then, M' satisfies the following claim.

Claim 3.2.5. *For every parameter $\delta^{-1} \in \mathbb{N}$, it holds that*

$$\Pr_{x\sim\mathcal{C}_n,M'}\left[n_x + 2 \le i^* \le n_x + \log\delta^{-1}\right] \ge 1 - 2(\ell(n)^2 + \ell(n)\log^2 n + 1)\cdot\delta$$

Proof of Claim 3.2.5. First, we show the upper bound. Fix x and each $h_{i,j}$ arbitrarily. For each i and j, the number of hash values of R_x, i.e., $|\{h_{i,j}(r)_{[i]} : r \in R_x\}|$ is at most $|R_x| = N_x$. Notice that M can find an inverse of $\langle h_{i,j}, i, x, v_{i,j}\rangle$ only if $v_{i,j}$ corresponds to one of the at most N_x hash values. Thus, M is successful with probability at most $N_x \cdot 2^{-i}$ over the choice of $v_{i,j} \sim \{0,1\}^i$. Particularly, for every $i > n_x + \log\delta^{-1}$ and every $j \in [\ell(n)]$, the success probability of M is at most

$$N_x \cdot 2^{-i} < 2^{n_x+1}\cdot 2^{-(n_x+\log\delta^{-1})} = 2\delta.$$

By the union bound, with probability at least $1 - 2\ell(n)(\ell(n) + \log^2 n)\delta$, the algorithm M fails to find an inverse for all $i \in [\ell(n) + \log^2 n]$ with $i > n_x + \log\delta^{-1}$ and for all $j \in [\ell(n)]$. In this case, it holds that $i^* \le n_x + \log\delta^{-1}$.

Next, we show the lower bound. Let $i = n_x + 2$. For readability, we omit parameters 1^n and $1^{(\ell(n)+\log^2 n)\gamma^2}$ for M below. Remember that

$$\Pr_{x,h,i',v}\left[\langle h, i', x, v\rangle \notin \mathrm{Im}E_n \vee E_n(M(\langle h, i', x, v\rangle)) = \langle h, i', x, v\rangle\right] \ge 1 - (\ell(n) + \log^2 n)^{-1}\gamma^{-2},$$

where $x \sim \mathcal{C}_n$, $h \sim \mathcal{H}_{\ell(n),\ell(n)+\log^2 n}$, $i' \sim [\ell(n) + \log^2 n]$, and $v \sim \{0,1\}^i$.

By Markov's inequality, with probability at least $1 - \gamma^{-1} \ge 1 - \delta$ over the choice of $x \sim \mathcal{C}_n$,

$$\Pr_{h,i',v}\left[\langle h, i', x, v\rangle \notin \mathrm{Im}E_n \vee E_n(M(\langle h, i', x, v\rangle)) = \langle h, i', x, v\rangle\right] \ge 1 - (\ell(n) + \log^2 n)^{-1}\gamma^{-1}.$$

Since i' corresponds to $i = n_x + 2$ with probability $(\ell(n) + \log^2 n)^{-1}$, we have

$$\Pr_{h,v}\left[\langle h, i, x, v\rangle \notin \mathrm{Im}E_n \vee E_n(M(\langle h, i, x, v\rangle)) = \langle h, i, x, v\rangle\right] \ge 1 - \gamma^{-1}.$$

We fix any $x \in \{0,1\}^n$ satisfying the above inequality. Remember that $R_x = \{r \in \{0,1\}^{\ell(n)} : D(r)_{|x|} = x\}$. For every $r, r' \in R_x$ with $r \ne r'$, the collision probability that $h(r)_{[i]} = h(r')_{[i]}$ is 2^{-i} over the choice of h. By the union bound, for every $r \in R_x$, the probability that there exists another $r' \in R_x \setminus \{r\}$ satisfying that $h(r)_{[i]} = h(r')_{[i]}$ is at most $|R_x|\cdot 2^{-i} = |R_x|\cdot 2^{-n_x-2} \le 2^{-1}$. For each $r \in R_x$ and $h \in \mathcal{H}_{\ell(n),\ell(n)+\log^2 n}$, let $A_{r,h}$ be a random variable that takes 1 iff there exists no $r' \in R_x \setminus \{r\}$ such that $h(r)_{[i]} = h(r')_{[i]}$ (otherwise, $A_{r,h} = 0$). Then, it holds that

$$\mathop{\mathbb{E}}_h\left[\sum_{x\in R_x} A_{x,h}\right] = \sum_{x\in R_x}\mathop{\mathbb{E}}_h[A_{x,h}] \ge \frac{N_x}{2}.$$

For each h, the number of hash values of R_x, i.e., $|\{h(r)_{[i]} : r \in R_x\}|$ is at least $\sum_{x\in R_x} A_{x,h}$. Thus, the random value $v \sim \{0,1\}^i$ corresponds to one of the hash values with probability at least $2^{-i}\cdot\sum_{x\in R_x} A_{x,h}$. Therefore, we have

$$\Pr_{h,v}[\langle h, i, x, v\rangle \in \mathrm{Im}E_n] \ge \mathop{\mathbb{E}}_h\left[2^{-i}\cdot\sum_{x\in R_x} A_{x,h}\right] \ge \frac{N_x}{2\cdot 2^i} \ge \frac{2^{n_x}}{2\cdot 2^{n_x+2}} = \frac{1}{8}$$

Since $\gamma \geq 16$, we obtain that

$$\begin{aligned}
&\Pr_{h,v}[E_n(M(\langle h,i,x,v\rangle)) \neq \langle h,i,x,v\rangle | \langle h,i,x,v\rangle \in \mathrm{Im}E_n] \\
&\quad = \Pr_{h,v}[\langle h,i,x,v\rangle \in \mathrm{Im}E_n \wedge E_n(M(\langle h,i,x,v\rangle)) \neq \langle h,i,x,v\rangle | \langle h,i,x,v\rangle \in \mathrm{Im}E_n] \leq 8\gamma^{-1} \leq 2^{-1},
\end{aligned}$$

and

$$\begin{aligned}
&\Pr_{h,v}[E_n(M(\langle h,i,x,v\rangle)) = \langle h,i,x,v\rangle] \\
&\quad \geq \Pr_{h,v}[\langle h,i,x,v\rangle \in \mathrm{Im}E_n] \cdot \Pr_{h,v}[E_n(M(\langle h,i,x,v\rangle)) = \langle h,i,x,v\rangle | \langle h,i,x,v\rangle \in \mathrm{Im}E_n] \\
&\quad \geq \frac{1}{8} \cdot \frac{1}{2} \geq \frac{1}{16}.
\end{aligned}$$

Since $(h_{i,j}, v_{i,j})$ is selected independently at random for each $j \in [\ell(n)]$, there is no $(h_{i,j}, v_{i,j})$ for which M is successful with probability at most $(1-1/16)^{\ell(n)} \leq 2^{-\Omega(\ell(n))}$. Notice that if this event does not occur, then $i^* \geq i = n_x + 2$. Thus, we conclude that

$$\Pr_{x\sim\mathcal{C}_n,M'}[i^* \geq n_x + 2] \geq (1-\delta)(1-2^{-\Omega(\ell(n))}) \geq 1 - 2\delta,$$

where we assume that $\delta \geq 2^{-\Omega(\ell(n))}$; otherwise, we can directly compute N_x and n_x by trying all $r \in \{0,1\}^{\ell(n)}$ in time $2^{\ell(n)} \leq \mathsf{poly}(\delta^{-1})$.

By the union bound, we have

$$\begin{aligned}
\Pr_{x\sim\mathcal{C}_n,M'}\left[n_x + 2 \leq i^* \leq n_x + \log \delta^{-1}\right] &\geq 1 - 2\ell(n)(\ell(n) + \log^2 n)\delta - 2\delta \\
&= 1 - 2(\ell(n)^2 + \ell(n)\log^2 n + 1)\delta.
\end{aligned}$$

This completes the proof of Claim 3.2.5. ◇

We construct the extrapolation algorithm Ext based on M and M': On input $x \sim \mathcal{C}_n, 1^{\langle k,\epsilon^{-1},\delta^{-1}\rangle}$, (i) Ext executes $i^* \leftarrow M'(x, 1^{2\delta^{-1}\gamma_0\gamma_1})$ and sets $\tilde{i} := i^* + 2\log\gamma_2 - 1$, where $\gamma_0 = 2(\ell(n)^2 + \ell(n)\log^2 n + 1)$, $\gamma_1 = 4\epsilon^{-1}$, and $\gamma_2 = \max\{8\epsilon^{-1}, 2\}$, (if M' outputs 0, then Ext outputs the empty symbol ε and halts); (ii) Ext picks $h \sim \mathcal{H}_{\ell(n),\ell(n)+\log^2 n}$ and repeats the following execution $m := 4\gamma_0\gamma_1\gamma_2^2\delta^{-1}\ell(n)$ times:

$$M(\langle\langle h, \tilde{i}, x, v\rangle, E_n\rangle, 1^n, 1^{2\delta^{-1}(\ell(n)+\log^2 n)\gamma_3})$$

for $r \sim \{0,1\}^{\tilde{i}}$ (where r is selected independently for each trial) and $\gamma_3 = 4\gamma_0\gamma_1\gamma_2^4\delta^{-1}$. If M outputs a valid inverse $(r', h, \tilde{i})$ of $\langle h, \tilde{i}, x, v\rangle$ (the validity can be easily verified), then Ext outputs $C(r')_{[|x|+1:|x|+k]}$ and halts; and (iii) if no inverse was found within the m trials, then Ext outputs the empty symbol ε and halts. It is not hard to verify that Ext halts in polynomial time.

We verify the correctness of Ext. For readability, we omit the unary parameters for Ext, M', and M.

By Claim 3.2.5,

$$\Pr_{x\sim\mathcal{C}_n,M'}\left[n_x + 2 \leq i^* \leq n_x + \log\delta^{-1}\gamma_0\gamma_1 + 1\right] \geq 1 - \frac{\delta}{2} \cdot 2(\ell(n)^2 + \ell(n)\log^2 n + 1)\gamma_0^{-1}\gamma_1^{-1} = 1 - \frac{\delta}{2}\gamma_1^{-1}$$

Thus, by Markov's inequality, the following holds with probability at least $1-\delta/2$ over the choice of $x \sim \mathcal{C}_n$:

$$\Pr_{M'}\left[n_x + 2 \leq i^* \leq n_x + \log \delta^{-1}\gamma_0\gamma_1 + 1\right] \geq 1 - \gamma_1^{-1} \tag{3.2}$$

Furthermore, on step (ii), we have

$$\Pr_{x,h,i,v}\left[\langle h,i,x,v\rangle \notin \mathrm{Im}E_n \vee E_n(M(\langle h,i,x,v\rangle)) = \langle h,i,x,v\rangle\right] \geq 1 - \frac{\delta}{2(\ell(n)+\log^2 n)} \cdot \gamma_3^{-1},$$

where $x \sim \mathcal{C}_n, h \sim \mathcal{H}_{\ell(n),\ell(n)+\log^2 n}$, $i \sim [\ell(n)+\log^2 n]$, and $v \sim \{0,1\}^i$. By Markov's inequality, the following holds with probability at least $1-\delta/2$ over the choice of $x \sim \mathcal{C}_n$:

$$\Pr_{h,i,v}\left[\langle h,i,x,v\rangle \notin \mathrm{Im}E_n \vee E_n(M(\langle h,i,x,v\rangle)) = \langle h,i,x,v\rangle\right] \geq 1 - \frac{1}{(\ell(n)+\log^2 n)} \cdot \gamma_3^{-1}.$$

In this case, it holds that

$$\forall i \in [\ell(n)+\log^2 n] \quad \Pr_{h,v}\left[\langle h,i,x,v\rangle \notin \mathrm{Im}E_n \vee E_n(M(\langle h,i,x,v\rangle)) = \langle h,i,x,v\rangle\right] \geq 1 - \gamma_3^{-1}, \tag{3.3}$$

because each i is selected with probabilitx $(\ell(n)+\log^2 n)^{-1}$.

By the union bound, with probability $1-\delta$ over the choice of $x \sim \mathcal{C}_n$, Eq. (3.2) and (3.3) are satisfied. For the theorem, it suffices to show that, under this condition,

$$\mathsf{L}_1\left(\mathsf{Ext}(x), \mathsf{Next}_k(\mathcal{D}_n, x)\right) \leq \epsilon.$$

Thus, we fix x that satisfies (3.2) and (3.3) arbitrarily and show the above.

On the execution of Ext, let B an event that M' does not output i^* satisfying that

$$n_x + 2 \leq i^* \leq n_x + \log \delta^{-1}\gamma_0\gamma_1 + 1$$

Then, we show the following claims:

Claim 3.2.6. $\Pr[B] \leq \epsilon/4$.

Claim 3.2.7. *Under the condition that $\neg B$, it holds that* $\mathsf{L}_1\left(\mathsf{Ext}(x), \mathsf{Next}_k(\mathcal{D}_n, x)\right) \leq 3\epsilon/4$

These claims imply Lemma 3.2.3 as

$$\mathsf{L}_1\left(\mathsf{Ext}(x), \mathsf{Next}_k(\mathcal{D}_n, x)\right) \leq 1 \cdot \Pr[B] + 3\epsilon/4 \cdot \Pr[\neg B] \leq \epsilon.$$

Thus, the remaining of the proof is to show Claims 3.2.6 and 3.2.7.

Proof of Claim 3.2.6. The claim immediately follows from Eq. (3.2) and $\gamma_1 = 4\epsilon^{-1}$. ◇

Proof of Claim 3.2.7. Under the condition $\neg B$, we have

$$n_x + 2\log\gamma_2 + 1 \leq \tilde{i} \leq n_x + 2\log\gamma_2 + \log\delta^{-1}\gamma_0\gamma_1.$$

For each $r \in R_x$ and $h \in \mathcal{H}_{\ell(n),\ell(n)+\log^2 n}$, we use the notation $A_{r,h}$ again to refer to the binary random variable that takes 1 iff there exists no $r' \in R_x \setminus \{r\}$ such that $h(r)_{[\tilde{i}]} = h(r')_{[\tilde{i}]}$. Then, for each $r \in R_x$, it holds that $\Pr_h[A_{r,h}=0] \leq N_x \cdot 2^{-\tilde{i}}$ by the union bound; thus,

$$\mathop{\mathbb{E}}_h\left[\sum_{r\in R_x} A_{r,h}\right] = \sum_{r\in R_x} \mathop{\mathbb{E}}_h\left[A_{r,h}\right] \geq N_x \cdot (1 - N_x 2^{-\tilde{i}}) \geq N_x \cdot (1 - 2^{n_x} 2^{-(n_x+2\log\gamma_2+1)}) \geq N_x \cdot (1 - \frac{1}{2\gamma_2^2}).$$

By Markov's inequality,

$$\Pr_h\left[\sum_{r\in R_x} A_{r,h} \geq (1-\frac{1}{\gamma_2})\cdot N_x\right] \geq 1-\frac{1}{2\gamma_2}.$$

Furthermore, by Eq. (3.3),

$$\Pr_{h,v}\left[\langle h,\tilde{i},x,v\rangle \notin \mathrm{Im}E_n \vee E_n(M(\langle h,\tilde{i},x,v\rangle)) = \langle h,\tilde{i},x,v\rangle\right] \geq 1-\gamma_3^{-1}.$$

By Markov's inequality, with probability at least $1-1/(2\gamma_2)$ over the choice of h,

$$\Pr_{v}\left[\langle h,\tilde{i},x,v\rangle \notin \mathrm{Im}E_n \vee E_n(M(\langle h,\tilde{i},x,v\rangle)) = \langle h,\tilde{i},x,v\rangle\right] \geq 1-2\gamma_2\gamma_3^{-1}. \tag{3.4}$$

Let $S_h = \{r\in R_x : A_{r,h}=1\}$ and $T_h = \{h(r)_{[\tilde{i}]} : r\in S_h\}$. Notice that $|S_h| = |T_h| = \sum_{r\in R_x} A_{r,h}$ holds. We call a hash function h satisfying Eq. (3.4) and $|T_h| \geq (1-\gamma_2^{-1})\cdot N_x$ a good hash function. By the union bound, h is good with probability at least $1-\gamma_2^{-1}$ over the choice of $h\sim\mathcal{H}_{\ell(n),\ell(n)+\log^2 n}$.

We fix a good hash function h arbitrarily. Then,

$$\Pr_{v\sim\{0,1\}^{\tilde{i}}}[v\in T_h] = |T_h|\cdot 2^{-\tilde{i}} \geq (1-\gamma_2^{-1})\cdot 2^{n_x}\cdot 2^{-(n_x+2\log\gamma_2+\log\delta^{-1}\gamma_0\gamma_1)} = (2\gamma_0\gamma_1\gamma_2^2)^{-1}\delta,$$

where we use the fact that $\gamma_2\geq 2$.

If $v\in T_h$, then $\langle h,\tilde{i},x,v\rangle \in \mathrm{Im}E_n$. Therefore, we have

$$\begin{aligned}&\Pr_v\left[E_n(M(\langle h,\tilde{i},x,v\rangle)) \neq \langle h,\tilde{i},x,v\rangle | v\in T_h\right]\\ &= \Pr_v\left[\langle h,\tilde{i},x,v\rangle \in \mathrm{Im}E_n \wedge E_n(M(\langle h,\tilde{i},x,v\rangle)) \neq \langle h,\tilde{i},x,v\rangle | v\in T_h\right] \leq 2\gamma_2\gamma_3^{-1}\cdot 2\gamma_0\gamma_1\gamma_2^2\delta^{-1} = \gamma_2^{-1}.\end{aligned}$$

Let $S'_h = \{r\in S_h : M(\langle h,\tilde{i},x,h(r)_{[\tilde{i}]}\rangle)) = \langle h,\tilde{i},x,h(r)_{[\tilde{i}]}\rangle\}$ and $T'_h = \{h(r)_{[\tilde{i}]} : r\in S'_h\}$. Then, we have $|S'_h| = |T'_h|$ and $|S'_h| \geq (1-\gamma_2^{-1})|S_h| \geq (1-\gamma_2^{-1})^2 N_x \geq (1-2\gamma_2^{-1})N_x$.

Under the condition that $v\in T'_h$, Ext obtains $r\in S'_h$ from M at uniformly random over S'_h. The total variation distance between the uniform distribution over S'_h and the uniform distribution over R_x is at most $2\gamma_2^{-1}$. Therefore, under the condition I_h that M finds a valid inverse $(r',h,\tilde{i})$ of $\langle h,\tilde{i},x,v\rangle$ (i.e., $D(r')_{[|x|]} = x$), the total variation distance between the conditional distribution $\tilde{\mathcal{D}}$ of r' and the uniform distribution over R_x (denoted by U_{R_x}) is bounded as follows:

$$\mathsf{L}_1(\tilde{\mathcal{D}}, R_x | I_h) \leq 2\gamma_2^{-1} + 1\cdot\Pr_v[v\notin T'_h | v\in\{h(r)_{[\tilde{i}]} : r\in R_x\}] \leq 2\gamma_2^{-1} + 2\gamma_2^{-1}N_x\cdot N_x^{-1} = 4\gamma_2^{-1} \leq \epsilon/2.$$

In this case,

$$\mathsf{L}_1(\mathsf{Ext}(x), \mathsf{Next}_k(\mathcal{D}_n,x)|I_h) = \mathsf{L}_1(D(\tilde{\mathcal{D}})_{[|x|+1:|x|+k]}, D(R_x)_{[|x|+1:|x|+k]}|I_h) \leq \mathsf{L}_1(\tilde{\mathcal{D}}, R_x|I_h) \leq \epsilon/2.$$

Next, we show that, for every good h, the algorithm M finds a valid inverse (i.e., the condition I_x is satisfied) at some trial with probability at least $1-2^{-\Omega(\ell(n))}$ over the choices of $v\sim\{0,1\}^{\tilde{i}}$. This implies the claim as

$$\begin{aligned}\mathsf{L}_1(\mathsf{Ext}(x), \mathsf{Next}_k(\mathcal{D}_n,x)) &\leq 1\cdot\Pr_h[h \text{ is not good}] + \mathsf{L}_1(\mathsf{Ext}(x), \mathsf{Next}_k(\mathcal{D}_n,x)|h \text{ is good}, I_h)\\ &\quad + 1\cdot\Pr[M \text{ finds no valid inverse for all } m \text{ trials}|h \text{ is good}]\\ &\leq \gamma_2^{-1} + \epsilon/2 + 2^{-\Omega(\ell(n))}\\ &\leq \epsilon/8 + \epsilon/2 + \epsilon/8 = 3\epsilon/4,\end{aligned}$$

where we assume that $\epsilon/8 \geq 2^{-\Omega(\ell(n))}$; otherwise, we can try all $r \in \{0,1\}^{\ell(n)}$ and perfectly simulate $\mathsf{Next}_k(\mathcal{D}_n, x)$ in time $2^{\ell(n)} \leq \mathsf{poly}(\epsilon^{-1})$.

Remember that

$$\Pr_v[v \in T_h] \geq (2\gamma_0\gamma_1\gamma_2^2)^{-1}\delta \text{ and } \Pr_v\left[E_n(M(\langle h, \tilde{i}, x, v\rangle)) = \langle h, \tilde{i}, x, v\rangle | v \in T_h\right] \geq 1 - \gamma_2^{-1} \geq 2^{-1}.$$

Therefore,

$$\begin{aligned}\Pr_v\left[E_n(M(\langle h, \tilde{i}, x, v\rangle)) = \langle h, \tilde{i}, x, v\rangle\right] &\geq \Pr_v\left[E_n(M(\langle h, \tilde{i}, x, v\rangle)) = \langle h, \tilde{i}, x, v\rangle | v \in T_h\right] \cdot \Pr_v[v \in T_h] \\ &\geq (4\gamma_0\gamma_1\gamma_2^2)^{-1} \cdot \delta.\end{aligned}$$

Therefore, M fails to find some inverse with probability at most $1 - (4\gamma_0\gamma_1\gamma_2^2)^{-1} \cdot \delta$. Thus, the probability that M finds no inverse in all m trials is at most

$$(1 - (4\gamma_0\gamma_1\gamma_2^2)^{-1} \cdot \delta)^m \leq (1 - (4\gamma_0\gamma_1\gamma_2^2)^{-1} \cdot \delta)^{4\gamma_0\gamma_1\gamma_2^2\delta^{-1}\ell(n)} \leq 2^{-\Omega(\ell(n))},$$

as desired. ◇

This completes the proof of Lemma 3.2.3. □

Chapter 4

Learning in Pessiland I: A Unified Theory of Average-Case Learning

In this chapter, we study which learning tasks is efficiently feasible in Pessiland to reveal a deep connection between learning theory and cryptography. Namely, we investigate learning-theoretic implications from the non-existence of a one-way functions.

4.1 Background

A one-way function (OWF) is a function that is easy to evaluate but hard to invert on average, and it is one of the most important cryptographic primitives. One importance of OWF lies in its robustness, i.e., the existence of OWF characterizes the existence of other various cryptographic primitives, such as a pseudorandom generator (PRG) [HILL99; VZ12; HRV13], a pseudorandom function (PRF) [GGM86], a digital signature [Rom90], a zero-knowledge proof [GMW91; Ost91; OW93; NOV06] and a private-key encryption scheme [GGM84]. In this sense, a rich theory of cryptography has been developed over decades under the axiom that OWF exists.

An opposite and less understood research direction is an algorithmic implication of the *non-existence* of OWF. This question was posed by Impagliazzo [Imp95], who asked whether there is a rich theory even under the axiom that there is no OWF. This research question is also fundamental because improving such an algorithmic consequence yields a characterization of OWF by weaker and more reliable hardness assumptions. For instance, the well-known open question of basing OWF on the hardness of NP is the ultimate goal of the research direction, where we need to show that the non-existence of OWF is sufficient to solve every NP problem.

Impagliazzo and Levin [IL90] first studied efficient *learning* under the non-existence of OWF and revealed a deep connection between two central fields in theoretical computer science, i.e., learning theory and cryptography. Particularly, Impagliazzo and Levin [IL90] introduced the notion of *universal extrapolation*, and it *seems* to solve *some* average-case tasks of learning efficiently. The reason why we use the vague words "*seems*" and "*some*" in the previous sentence is that they specified neither which learning tasks can be solved by their ideas nor a concrete statement of what they called *universal extrapolation*. In fact, their notion of learning appears very different from standard models in learning theory, such as the PAC learning model introduced by Valiant [Val84].

Since the work [IL90] introduced universal extrapolation and hinted at the connection between learning theory and cryptography, several follow-up works have investigated learning-theoretic im-

plications from the non-existence of OWF in more popular learning models. Blum, Furst, Kearns, and Lipton [BFKL93] introduced a natural average-case variant of the PAC learning model and showed that polynomial-sized circuits are learnable on average under the non-existence of OWF. In their model, a learner is given a sample set $(x^1, f(x^1)), \ldots, (x^{\mathsf{poly}(n)}, f(x^{\mathsf{poly}(n)}))$, where each $x^i \in \{0,1\}^n$ is identically and independently drawn from a *fixed* samplable example distribution $\mathcal{D}$, and f is an unknown target circuit selected according to a *fixed* samplable distribution $\mathcal{F}$. Here, by "fixed distributions", we mean that the distributions are fixed beforehand, and the learner $L_{\mathcal{D},\mathcal{F}}$ can depend on the distributions $\mathcal{D}$ and $\mathcal{F}$. The task of the learner is to output a good hypothesis h that approximates f well under the same example distribution $\mathcal{D}$ with high probability over the choices of $f, x^1, \ldots, x^{\mathsf{poly}(n)}$ and randomness for the learner. In fact, Impagliazzo [Imp95] explained the unspecified learning-theoretic consequence of [IL90] in this form. Another work by Naor and Rothblum [NR06] showed that learning *adaptively changing distributions* (ACDs) is feasible under the non-existence of OWF. Their learning setting is the following: (i) at the initial step, a hidden initial state $s := s_0 \in \{0,1\}^{\mathsf{poly}(n)}$ is selected according to a *fixed* samplable distribution $\mathcal{G}$; (ii) at each i-th stage, a sample x^i is generated by a *fixed* polynomial-time sampler D as $x^i \leftarrow D(s; r)$, where r is a hidden random seed, and D can update its internal state s according to r; (iii) a learner is given a stream of the samples $x^1, x^2, \ldots$, and the task of the learner is to choose a stage i and to approximately simulate the conditional distribution of the next outcome x^i given the initial state s_0 and the past stream $x^1, \ldots, x^{i-1}$ at stage i without observing x^i. Later, Naor and Yogev [NY19] studied the relationships between OWF and a Bloom filter in an adversarial setting. Here, the Bloom filter is a space-bounded and efficiently computable filtering algorithm of which no efficient adversary can find a false positive during adaptive interactions. By using the learner for ACDs, they showed that for any nontrivial Bloom filter, there exists an adversary that finds a false positive under the non-existence of OWF, i.e., a false positive of a *fixed* Bloom filter is efficiently learnable by adaptive interactions.

These follow-up works no longer used the original idea of universal extrapolation, and the relationships between their learning methods and the original idea of universal extrapolation remain unclear. Furthermore, the learners in the follow-up works are distribution specific in the sense that the underlying samplable distributions are fixed beforehand. Namely, the learners need to know the underlying samplable distribution. This is undesirable in learning because usually, we have no way to know the underlying distribution that generates data before the learning process starts.

4.2 Our Results

In this work, we present a general framework for constructing various learning algorithms that work in average-case settings under the non-existence of OWF. The framework unifies, generalizes, simplifies, and improves the currently known learning algorithms that follow from the non-existence of OWF. Surprisingly, the framework is based on *universal extrapolation*. Namely, the very first idea described by the pioneering work of Impagliazzo and Levin [IL90], if interpreted correctly, improves the learnability results that have been discovered over the last decades! For instance, the following learnability results are obtained from universal extrapolation.

Theorem 4.2.1 (Universal learning ACDs, informal)**.** *If there exists no infinitely-often one-way function, then there exists an efficient randomized algorithm that learns every* unknown *ACD.*

It is worthy of note that learning *unknown* ACDs was thought to be *information-theoretically impossible* in the previous work [NR06]. Therefore, in Theorem 4.2.1, we obtain a one-way function

whose security is based on the intractability of a seemingly impossible average-case learning. We will elaborate on this point later in this section.

Theorem 4.2.2 (Universal agnostic learning on average, informal)**.** *If there exists no infinitely-often one-way function, then polynomial-length multi-output functions are efficiently agnostic learnable on average when samples are independently and identically drawn from an unknown sampling algorithm with $s(n) = \mathsf{poly}(n)$-bit secret advice that is selected according to another unknown samplable distribution. Furthermore, the sample complexity is $O(s(n)\epsilon^{-2})$ for any accuracy parameter $\epsilon \in (0, 1]$.*

Agnostic learning was introduced by Kearns, Schapire, and Sellie [KSS94] as a generalization of PAC learning. In agnostic learning, a learner is given samples of the form (x, b) (we call x and b an *example* and a *label* of x, respectively) that are selected identically and independently according to an unknown distribution. The difference from PAC learning is that the label b can also be selected according to some distribution determined by x. The goal of the learner for a target class $\mathscr{C}$ of functions is to find a good hypothesis that approximates (within an additive accuracy parameter ϵ) the best function $f \in \mathscr{C}$ that approximates the labels[1]. In Theorem 4.2.2, we introduce a natural average-case variant of agnostic learning, where we consider the average-case performance over the choice of $\mathsf{P/poly}$-samplable distributions on samples. Note that the well-known learning problems *learning parity with noise* (LPN) and *learning with errors* (LWE) are special cases of our agnostic learning model, where labels are computed by a linear function and affected by classification noise independently of examples, and the target class $\mathscr{C}$ is the class of linear functions. Our model includes more general noise distributions that depend on examples. Furthermore, we consider the entire class of functions as the target class $\mathscr{C}$, i.e., our learner can approximate the best function that predicts labels in all (possibly not efficiently computable) functions. This is impossible in the original (worst-case) agnostic learning model because a function that computes labels is possibly not efficiently approximated. We also remark that the sample complexity $O(s(n)\epsilon^{-2})$ (i.e., the number of samples required for learning) is optimal with respect to $s(n)$ and ϵ^{-1} (see Section 4.9.4).

The learnability results in Theorems 4.2.1 and 4.2.2 yield characterizations of OWF because efficient learners in Theorems 4.2.1 and 4.2.2 are sufficient to break the security of a cryptographic primitive called a pseudorandom function family, and the existence of a pseudorandom function family and the existence of OWF are equivalent [GGM86; HILL99]. In addition, by the well-known reduction from breaking the security of a pseudorandom function family to weak learning with membership queries under the uniform example distribution [Val84], our results establish the robustness of average-case learning.

Corollary 4.2.3. *The following are equivalent:*

- *The non-existence of infinitely-often one-way function;*
- *Average-case weak learning for $\mathsf{P/poly}$ with membership queries under a uniform example distribution and a known samplable distribution that selects a target function;*
- *Average-case universal agnostic learning (as in Theorem 4.2.2);*
- *Universal learning ACDs (as in Theorem 4.2.1); and*

[1] Strictly speaking, we focus on agnostic learning for the 0-1 loss (i.e., the prediction loss in [KSS94]) in this work. Note that our learner can be generalized to the case of general (polynomial-time computable) loss functions if the number of labels is polynomially bounded (see also Sections 4.3.3 and 4.9.3).

- *Average-case universal distributional learning.*

We mean by *average-case universal distributional learning* the special case of learning ACDs, where the internal state never changes, i.e., samples are drawn independently and identically from an unknown P/poly-samplable distribution $\mathcal{D}$, and $\mathcal{D}$ is selected from another unknown samplable distribution $\mathcal{G}$. This is a natural average-case variant of distributional learning introduced in [KMRRSS94].

In Corollary 4.2.3, we can observe many surprising phenomena, such as (i) a reduction from agnostic learning to PAC learning, (ii) a reduction from universal learning to distribution-specific learning (particularly, under the uniform example distribution), (iii) boosting of accuracy, (iv) a reduction from learning distributions to learning binary classification, (v) a reduction from learning with membership queries to learning with random samples. Furthermore, Corollary 4.2.3 yields the first equivalence between the existence of OWF and average-case hardness of agnostic learning. To the best of our knowledge, such a characterization of OWF by hardness of learning in noisy settings was not known before even in the restricted case of learning with independent random noise.

Before presenting the relationships between the results above and universal extrapolation, we highlight the differences from the previous results and the difficulty of the improvements.

Theorem 4.2.1 strengthens the result of [NR06] in that the learner is universal and does not need the description of the underlying distribution, i.e., a samplable distribution $\mathcal{G}$ of the initial state and a polynomial-time sampler D that generates samples and updates the internal state. We remark that making their learner universal is very challenging. In fact, Naor and Rothblum [NR06] explicitly mentioned that their learner requires the description of $(\mathcal{G}, D)$ and even claimed that universal learning ACDs is *information-theoretically impossible*. Their arguments are as follows. An algorithm for learning ACDs *must* use its knowledge of $(\mathcal{G}, D)$ to decide, on-the-fly, at what stage it tries to simulate the next outcome (remember that a learner can select a prediction stage by itself) because (for example) an unknown $(\mathcal{G}, D)$ can keep specific $i \in \mathbb{N}$ hidden from the universal learner, and D may output some fixed distribution up to round i, and only then begin to use its secret state[2]. However, our result falsifies their claim. Our learner predicts successfully for most stages i and thus decides the prediction stage *uniformly at random* without using any knowledge of $(\mathcal{G}, D)$. We also improve the dependence of a (confidence) parameter in sample complexity (see Theorem 4.8.5). Note that, by replacing [NR06] in [NY19] with Theorem 4.2.1, we also strengthen the result [NY19] in the sense that we construct a "universal" adversary that breaks all *unknown* nontrivial Bloom filters if there is no OWF. Namely, even if we can hide the algorithms of Bloom filters from adversaries, OWF is necessary for a nontrivial Bloom filter in the adversarial setting.

Theorem 4.2.2 strengthens the result of [BFKL93] (i.e., the average-case easiness of PAC learning) in the following four points: (i) the label is generalized from binary to polynomial length; (ii) the framework is generalized from PAC learning to agnostic learning and (iii) from distribution-specific to universal, and (iv) the applicable settings of distributions on samples are broadened. Particularly, the work [BFKL93] only considered the case in which a target function is selected independently of the example distribution, while our learner works on average for randomly selected P/poly-samplable joint distributions over samples. Note that the previous learner in [BFKL93] obtained from the non-existence of OWF does not work in the general case of joint distributions unless we solve the notorious open problem of basing OWF on the average-case hardness of NP (i.e., excluding Pessiland). This is because (i) the learner in [BFKL93] is a proper learner, where

[2]Even trying all possibilities of $(\mathcal{G}, D)$ does not yield universal learning ACDs because it produces many hypotheses but there seems to be no way to verify the performance of the hypothesis in universal learning ACDs (see Section 4.3.5).

the learner outputs a hypothesis in the same class of target functions, and (ii) the NP-hardness result of proper learning [PV88] implies that every distributional NP problem is reducible to proper learning 2-term DNFs in our average-case setting. We bypass this difficulty by considering improper learning.

4.3 Techniques: Universal Extrapolation, Revisited

Now, based on universal extrapolation, we present the general framework for constructing learners that work in average-case settings. Universal extrapolation has the capability for improving the previous learnability results as seen in Section 4.2. In addition, one of the main theorems of the recent work [IRS22] easily follows from a proposition for universal extrapolation [IL90, Proposition 1]. Nevertheless, its significance has been overlooked for decades in the theoretical computer science (TCS) community. First, we review what was known for universal extrapolation and why the significance has been overlooked.

4.3.1 Universal Extrapolation Outlined in [IL90]

The main reason why the significance of universal extrapolation has been overlooked for decades is that it was only hinted at so abstractly that researchers could not understand the outlined ideas and study their consequences. The ideas outlined in [IL90] are described as follows.

1. Under the non-existence of OWF, there exists a polynomial-time algorithm that is given a string x selected according to any *known* samplable distribution $\mathcal{D}$ and approximates the probability $\mathcal{D}(x)$ that x is drawn from $\mathcal{D}$ well on average ([IL90, Lemma 1]).

2. By applying the algorithm in step 1 to the distribution Q^t of outcomes of a time-bounded universal Turing machine U^t, we obtain a *universal* approximation algorithm that is given a string x selected according to an *unknown* samplable distribution $\mathcal{D}$ and approximates the probability $\mathrm{Q}^t(x)$ that x is drawn from Q^t in polynomial time on average ([IL90, Proposition 1]).

3. By applying the algorithm in step 2, they approximate the likelihood of the bit subsequent to x under Q^t, where x is selected according to an *unknown* samplable distribution $\mathcal{D}$. Then, they apply Solomonoff's inductive inference [Sol64a; Sol64b] in a time-bounded setting, and "it yields *the universal extrapolation*" (but no formal statement can be found in [IL90]).

We remark that the ideas above seem to be lost in the current TCS community. For example, even the intermediate step 2 implicitly contains the recent important result [IRS22] in meta-complexity that established a new characterization of the existence of OWF by the average-case hardness of approximating Kolmogorov complexity, which follows from the known fact that the value of $-\log \mathrm{Q}^t(x)$ approximates the Kolmogorov complexity of x well on average.[3] In addition, verifying the correctness of the ideas above is also hard because they presented only a high-level proof sketch of step 1, and there is no proof sketch of steps 2 and 3, except an abstract introduction of Solomonoff's inductive inference. Furthermore, understanding the proof sketch of step 1 is also difficult. Although several recent studies [Nan21b; IRS21] tried to formally prove step 1, what

[3] In [IRS22, page 1580], the authors discussed the reason why their results were not shown earlier. Surprisingly, we show that the algorithm implicit in [IL90, Proposition 1] gives an algorithm better than [IRS22]. See Section 4.6.

they showed is worse approximation factors than the original lemma in [IL90]. Other important studies [OW93; ABX08; LP20] also mentioned universal extrapolation as related work or a lemma for showing main theorems, but the statements vary and seem weaker than the original ideas above; for instance, [OW93] stated universal extrapolation as a distribution-specific algorithm that extrapolates a given prefix under a known samplable distribution. However, this directly follows from the non-existence of distributional inverters [IL89] and seems to lose the universality. As seen above, the TCS community has no longer known what universal extrapolation is, let alone the algorithmic consequences.

The main contribution of this work is to formalize the original ideas of [IL90] and make them available in the TCS community. Particularly, we first revisit the formal statement of universal extrapolation and then investigate the general framework to translate universal extrapolation into various learning algorithms. Then, the results in Section 4.2 are obtained as special cases.

4.3.2 Formulating Universal Extrapolation

We fix a universal Turing machine U arbitrarily. For every $t \in \mathbb{N}$, we use the notation U^t to refer to the execution of U in t steps[4]. Then, U gives rise to the following important distribution.

Definition 4.3.1 (Universal distribution)**.** *For each $t \in \mathbb{N}$, the t-time-bounded universal distribution Q^t is defined as the distribution of the output of $U^t(r)$ for a uniformly random seed $r \sim \{0,1\}^t$.*

In this study, we formulate universal extrapolation as a randomized algorithm UE that, for a given $k \in \mathbb{N}$ and a prefix x of a string drawn from an *unknown* samplable distribution, extrapolates the next k bits subsequent to x under the time-bounded universal distribution Q^t. We construct UE under the non-existence of OWF.

Theorem 4.3.2 (Universal extrapolation)**.** *If there exists no infinitely-often one-way function, then there exists a randomized polynomial-time algorithm UE such that for every samplable distribution $\mathcal{D} = \{\mathcal{D}_n\}_{n\in\mathbb{N}}$ over binary strings, for all large enough $n, t \in \mathbb{N}$, and for all $i, k, \epsilon^{-1}, \delta^{-1} \in \mathbb{N}$,*

$$\Pr_{x\sim\mathcal{D}_n}\left[\mathsf{L}_1\left(\mathsf{UE}(x_{[i]}; 1^{\langle n,k,t,\epsilon^{-1},\delta^{-1}\rangle}), \mathsf{Next}_k(\mathrm{Q}^t, x_{[i]})\right) \leq \epsilon\right] \geq 1-\delta.$$

Here, n must be larger than $2^d = O(1)$, where d is the description length of the sampling algorithm for $\mathcal{D}$, and t must be polynomially larger than the time complexity of $\mathcal{D}_n$ (see Theorem 4.7.1 for a more precise statement about parameters).

In this work, we formally prove Theorem 4.3.2 above by two different constructions of UE. The first one is based on the well-known equivalence between distributional OWF and OWF [IL89], and the second one is based on a universal approximation algorithm of the universal a priori probability. The former is simpler in the sense that it extrapolates the next bits simultaneously, and it is analogous to the extrapolation algorithms that have been considered for decades in TCS [Ost91; OW93; NR06; ABX08; Xia10; NY19]. The latter extrapolates next k bits inductively one by one, and this seems to be the original intention in [IL90][5].

[4] More precisely, we assume that each Turing machine has a write-only output tape, and we do not take into account whether it halts or not when we consider U^t.

[5] It is unclear whether extrapolating multiple bits was the original intention in [IL90] because the inductive generation based on inverting functions usually causes the super-polynomial blowup of computational time for extrapolating the next $\omega(1)$ bits.

4.3.3 Translating Universal Extrapolation into Learning

Next, we investigate the relationship between UE in Theorem 4.3.2 and learning algorithms. Particularly, we formulate the idea outlined in [IL90] of applying Solomonoff's inductive inference in a time-bounded setting. In this section, we present the general framework for translating UE into learning algorithms and verify that the results in Section 4.2 are obtained as special cases. Then, in Section 4.3.4, we explain why the framework works based on Solomonoff's inductive inference.

We consider the following general framework of online learning. At each discretized stage $i \in \mathbb{N}$, a learner first observes a string $x^i \in \{0,1\}^*$ (which represents auxiliary information for the i-th prediction) and then obtains another string $y^i \in \{0,1\}^*$ (which represents the i-th result), where x^i (resp. y^i) can be correlated with previous observations $x^1, y^1, \ldots, x^{i-1}, y^{i-1}$ (resp. $x^1, y^1, \ldots, x^{i-1}, y^{i-1}, x^i$). At stage i, the task of the learner is to predict the next outcome y^i from x^i and the previous observations $x^1, y^1, \ldots, x^{i-1}, y^{i-1}$ before observing y^i.

The general framework above captures many learning problems, including PAC learning, agnostic learning, and learning ACDs. In PAC learning, each x^i is an example independently and identically drawn from an example distribution $\mathcal{D}$, and each y^i is $f(x^i)$, where f is a target function. At some stage m, an input x^m to the hypothesis is drawn from $\mathcal{D}$, and the goal of the learner is to output b as the prediction of $y^m = f(x^m)$. In this case, the prediction stage m represents the number of samples required for PAC learning (i.e., sample complexity). Agnostic learning is also captured in the same way except that each y^i is also selected according to some distribution determined by x^i. In learning ACDs, each x^i is an empty string ε, and each y^i is the i-th sample drawn from ACD. At some stage m, the learner tries to statistically simulate the distribution of the next outcome y^m. Note that, as seen in the examples above, the stage m at which a learner makes some prediction generally captures the sample complexity required for learning.

Now, we consider a *cheating*[6] learner L^* that predicts the next outcome y^i at step i *with additional access to oracle* that returns a sample selected according to the distribution of y^i (i.e., the conditional distribution given previous observations $x^1, y^1, \ldots, x^{i-1}, y^{i-1}, x^i$) for each access. Note that the additional oracle access is powerful and suffices for the learning tasks given as examples above. In PAC learning and learning ACDs, it is enough to directly output a sample obtained from the oracle, and it trivially simulates the distribution of the next outcome. The case of agnostic learning is less obvious, but it suffices to obtain enough labels from the oracle and outputs the label generated most frequently.

The general framework for translating universal extrapolation to learning algorithms is the following: for any cheating learner L^*, if we construct a (standard) learner L *by replacing the oracle with universal extrapolation* UE given $x^1, y^1, \ldots, x^{i-1}, y^{i-1}, x^i$ as input, then the behavior of L is not much different from the one of L^* *on average* as long as the total number m of stages (i.e., sample complexity) is large enough. Namely, universal extrapolation enables us to reduce constructing a learner for some task to constructing a *cheating* learner for the same task. Below, we present the details, particularly, we explain the intention of the term "on average" and how large the total number m of stages should be.

We consider the following average-case setting on the framework above: (i) the offline stream of data $x^1, y^1, \ldots, x^m, y^m$ is selected according to an *unknown* P/poly-samplable distribution, i.e., a sampling algorithm with advice z of $s(n) = \mathsf{poly}(n)$ bit (we regard this z as secret information), and (ii) the secret advice z is selected according to an *unknown* samplable distribution. For instance, in

[6]The term "cheating" is due to the fact that the learner can freely read the future by the oracle without making any prediction from the past.

Theorem 4.2.1, the secret information z corresponds to the initial state s_0, and in Theorem 4.2.2, the secret information z corresponds to the secret advice for generating samples. Then, the learner L obtained from UE performs as well as the corresponding cheating learner L^* with high probability over the choice of z, a random position $i \sim [m]$, and the choice of past data $x^1, y^1, \ldots, x^{i-1}, y^{i-1}$ as long as m is sufficiently large. This is stated as the following meta-theorem, where the requirement of m is also indicated. For a more detailed statement, see Theorem 4.8.1.

Theorem 4.3.3. *For a randomized cheating learner $L^{*,?}$ of query complexity $q(\cdot)$, let L be the learner obtained by simulating the oracle access by UE (with a proper choice of parameters of UE).*

Then, for every samplable distribution $\mathcal{G} = \{\mathcal{G}_s\}_{s\in\mathbb{N}}$ and every P/poly-samplable distribution $\mathcal{D} = \{\mathcal{D}_z\}_{z\in\{0,1\}^*}$, where each $\mathcal{G}_s$ is over s-bit strings, and each $\mathcal{D}_z$ is over offline streams $x^1, y^1, \ldots, x^m, y^m$ (for $m := m(z)$), for every large enough $s, t \in \mathbb{N}$, every $\delta^{-1}, \lambda^{-1} \in \mathbb{N}$, and every auxiliary input $w \in \{0,1\}^*$, if $m \geq m_0 = O(s \cdot q(w) \cdot \delta^{-1}\lambda^{-2})$, then

$$\Pr_{z,i,x^1,\ldots,y^{i-1},x^i}\left[\mathsf{L}_1\left(L(x^1, y^1 \ldots, x^{i-1}, y^{i-1}, x^i, w; 1^{\langle r,t,\delta^{-1},\lambda^{-1}\rangle}), L^{*,\mathcal{D}_z^{y^i}}(w)\right) \leq \lambda\right] \geq 1 - \delta,$$

where $z \sim \mathcal{G}_s, i \sim [m]$, $x^1, y^1 \ldots, x^{i-1}, y^{i-1}, x^i$ are selected according to $\mathcal{D}_z$, and $\mathcal{D}_z^{y^i}$ is an oracle that returns a sample drawn from the conditional distribution of y^i given $x^1, y^1 \ldots, x^{i-1}, y^{i-1}, x^i$ with respect to $\mathcal{D}_z$.

In words, if the total number m of steps is sufficiently large with respect to the amount s of secret information z and the query complexity q of the cheating learner L^*, then UE yields good statistical simulation of the cheating learner L^*. The parameter t is used for determining the time bound t of Q^t simulated by UE and must be polynomially larger than the time complexity of the sampling algorithm for offline streams. By applying Theorem 4.3.3 to the trivial 1-query cheating learner that directly outputs a sample from the oracle, we obtain Theorem 4.2.1 and average-case PAC learnability results, e.g., the setting in [BFKL93] and Theorem 4.2.2 with the additional promise that a label b is determined only by an example x.

Theorem 4.3.3 shows that every polynomial-time cheating learner can be statistically simulated by UE on average for any sufficiently large polynomial m because any polynomial-time cheating learner can make only polynomially many queries. However, m depends on the number of queries q in Theorem 4.3.3, and if the query complexity of the cheating learner depends on some parameters for learning, Theorem 4.3.3 may cause bad dependence of the parameter on sample complexity. For instance, in agnostic learning with an accuracy parameter $\epsilon \in (0,1)$, the learner tries to find a hypothesis h that approximates the best function for computing labels with the additive error ϵ over the choice of a sample. To implement this by the cheating learner that outputs the most frequent label, the cheating learner needs at least $q = \Omega(\epsilon^{-2})$ samples for empirical estimation to determine the most frequent label within an additive error $\Theta(\epsilon)$. As a result, the learner obtained by Theorem 4.3.3 requires at least $\Omega(q \cdot \epsilon^{-1}) = \Omega(\epsilon^{-3})$ samples (where the additional factor ϵ^{-1} comes from the guarantee on x^i), which is worse than the optimal dependence ϵ^{-2} in Theorem 4.2.2.

For agnostic learning with sample complexity of the optimal dependence ϵ^{-2}, we show another meta-theorem. Note that the idea of replacing the oracle with UE is the same, and the only difference is the analysis of the total number m. Particularly, for the second bound on m, we restrict the task of cheating learners to minimizing expected loss, as discussed below. Instead, we obtain the lower bound of m that does not depend on the query complexity of the cheating learner, and it yields better sample complexity in agnostic learning than the application of Theorem 4.3.3.

We consider the following specific task of cheating learners: Let $b \in \mathbb{N}$ be the length of each label y^i. At stage i, (i) a learner is given a past stream $x^{<i} := x^1, y^1 \dots, x^{i-1}, y^{i-1}$ and advice information x^i and chooses an action $a_{x^{<i},x_i}$ from a set $\mathcal{A}$ of actions; and (ii) the goal of the learner is to minimize the expected loss with respect to a loss function $l\colon \mathcal{A} \times \{0,1\}^b \to [0, C]$ (where $C > 0$ is constant), i.e., the learner tries to minimize

$$\mathop{\mathbb{E}}_{x^i,y^i}[l(a_{x^{<i},x_i}, y^i)].$$

For instance, agnostic learning discussed above is captured as the case in which the action set is $\{0,1\}^b$ (i.e., the label set), and a loss function $l : \{0,1\}^b \times \{0,1\}^b \to [0,1]$ is the 0-1 loss function defined as

$$l(a, y) = \begin{cases} 0 & \text{if } a = y \\ 1 & \text{if } a \neq y, \end{cases}$$

because

$$\mathop{\mathbb{E}}_{x^i,y^i}[l(a_{x^{<i},x_i}, y^i)] = \Pr_{x^i,y^i}[a_{x^{<i},x_i} \neq y^i].$$

Note that, in this case, the upper bound C of the loss function is 1.

The second meta-theorem is stated as follows, where we show the bound on m based on C. For a more detailed statement, see Theorem 4.9.2.

Theorem 4.3.4. *Let $b \in \mathbb{N}$. Let $\mathcal{A}$ be a set of actions, and let $l\colon \mathcal{A} \times \{0,1\}^b \to \mathbb{R}_{\geq 0}$ be a loss function satisfying that there exists $C > 0$ such that $l(a, y) \leq C$ for every $a \in \mathcal{A}$ and $y \in \{0,1\}^b$. For a cheating learner $L^{*,?}$ that outputs an action in $\mathcal{A}$, let L be the learner obtained by simulating the oracle access by* UE *(with a proper choice of parameters of* UE*).*

Then, for every samplable distribution $\mathcal{G} = \{\mathcal{G}_s\}_{s\in\mathbb{N}}$ and every P/poly*-time samplable distribution $\mathcal{D} = \{\mathcal{D}_z\}_{z\in\{0,1\}^*}$, where each $\mathcal{G}_s$ is over s-bit strings, and each $\mathcal{D}_z$ is over offline streams $x^1, y^1, \dots, x^m, y^m$ (for $m := m(z)$), for every large enough $s, t \in \mathbb{N}$, every $\epsilon^{-1}, \delta^{-1} \in \mathbb{N}$, and every auxiliary input $w \in \{0,1\}^*$, if $m \geq m_0 := O(s \cdot C^2 \cdot \epsilon^{-2}\delta^{-2})$, then*

$$\Pr_{z,i,x^{<i}}\left[\mathop{\mathbb{E}}_{x^i,y^i,L}\left[l(L(x^{<i}, x^i, w; 1^{\langle n,r,b,t,\epsilon^{-1},\delta^{-1}\rangle}), y^i)\right] \leq \mathop{\mathbb{E}}_{x^i}\left[\min_{\alpha\in\mathcal{A}} \mathop{\mathbb{E}}_{y^i}[l(\alpha, y^i)]\right] + 2\Delta_{L^*} + \epsilon\right] \geq 1 - \delta,$$

where $z \sim \mathcal{G}_s, i \sim [m]$, $x^{<i} := x^1, y^1 \dots, x^{i-1}, y^{i-1}$ is selected according to $\mathcal{D}_z$, and x^i, y^i are selected according to the conditional distribution of the i-th observation given $x^{<i}$ under $\mathcal{D}_z$, and

$$\Delta_{L^*} := \Delta_{L^*}(w, b) = \sup_{\mathcal{O}:\textit{distribution over } \{0,1\}^b} \left(\mathop{\mathbb{E}}_{\mathcal{O},y\sim\mathcal{O}}[l(L^{*,\mathcal{O}}(w), y)] - \min_{\alpha\in\mathcal{A}} \mathop{\mathbb{E}}_{y\sim\mathcal{O}}[l(\alpha, y)]\right).$$

In words, if the total number m of stages is sufficiently large with respect to the amount s of secret information z and the square of the upper bound C of the loss function, then L performs in the same quality as the cheating learner up to an additive error. For agnostic learning with accuracy parameter ϵ (i.e., Theorem 4.2.2), we first construct a cheating learner that estimates the most frequent label (i.e., approximates the best predictor) within the additive estimation error $\epsilon/4$ by collecting sufficiently many labels (i.e., $\Delta_{L^*} \leq \epsilon/4$), and then we apply Theorem 4.3.4 with the additive error $\epsilon/2$. As a result, we obtain an agnostic learner that outputs a hypothesis within accuracy error ϵ with high probability[7] on average with sample complexity $O(s(n) \cdot \epsilon^{-2})$. Note

[7]We can amplify the success probability to $1 - \delta$ for a given confidence parameter δ^{-1} by the standard repetition technique [HKLW88] with multiplicative factor $\log \delta^{-1}$ on the time and sample complexity.

that Theorem 4.3.4 also yields an efficient agnostic learner for a general loss function l as long as l is efficiently computable, and minimizing the expected loss with respect to l is efficiently solvable by a cheating learner. For instance, average-case universal agnostic learning is feasible for every polynomial-time computable loss function l under the non-existence of OWF when the total number of labels is polynomially bounded (see Section 4.9.3).

4.3.4 Why Do the Frameworks Work?

We present an overview of the proofs of Theorems 4.3.3 and 4.3.4. For every distributions $\mathcal{D}$ and $\mathcal{E}$, let $\mathrm{KL}(\mathcal{D}||\mathcal{E})$ represent the KL divergence between $\mathcal{D}$ and $\mathcal{E}$.

The following is the key lemma to show Theorem 4.3.3.

Lemma 4.3.5. *There exists a polynomial τ such that for every distribution $\mathcal{D}$ over $x^1, y^1, \ldots, x^m, y^m$, if $\mathcal{D}$ has a t_D-time sampler described by d bits, then for every $t, q \in \mathbb{N}$ with $t \geq \tau(d, t_D)$, and every q-query (possibility not efficiently computable) oracle algorithm I,*

$$\mathop{\mathbb{E}}_{i\sim[m],x^1,y^1,\ldots,x^{i-1},y^{i-1},x^i}\left[\mathrm{KL}\left(I^{\mathsf{Next}_{|y^i|}(\mathcal{D},x^1\cdots y^{i-1}x^i)}||I^{\mathsf{Next}_{|y^i|}(\mathrm{Q}^t,x^1\cdots y^{i-1}x^i)}\right)\right] \leq q\cdot\frac{O(d)}{m}.$$

Theorem 4.3.3 is obtained as a corollary to Lemma 4.3.5. Particularly, if I is a cheating learner, then the first distribution $I^{\mathsf{Next}_{|y^i|}(\mathcal{D},x^1\cdots y^{i-1}x^i)}$ is the distribution of the outcome of the cheating learner. While, the second distribution of $I^{\mathsf{Next}_{|y^i|}(\mathrm{Q}^t,x^1\cdots y^{i-1}x^i)}$ is the distribution of the outcome of the cheating learner, where the oracle is replaced with the extrapolation under Q^t. By choosing parameters for UE properly, UE statistically simulates Q^t as in Theorem 4.3.2. Therefore, for every sufficiently large m with respect to the query complexity q and the description size d of the sampler, we can show that the cheating learner is statistically simulated by replacing the oracle access with UE by a standard probabilistic argument.

The proof of Lemma 4.3.5 is based on Solomonoff's theory of inductive inference [Sol64a; Sol64b; LV19]. In the special case in which (i) I is the trivial 1-query algorithm that directly outputs a sample and (ii) $x^i = \varepsilon$ (the empty string) for all i, by careful calculations in Solomonoff's inductive inference, we can show that the expectation in Lemma 4.3.5 is bounded above by

$$\frac{1}{m}\mathop{\mathbb{E}}_{y^1,\ldots,y^m\sim\mathcal{D}}\left[\ln\frac{\mathcal{D}(y^1,\ldots,y^m)}{\mathrm{Q}^t(y^1,\ldots,y^m)}\right],$$

where $\mathcal{D}(y^1,\ldots,y^m)$ (resp. $\mathrm{Q}^t(y^1,\ldots,y^m)$) represents the probability that $(y^1,\ldots,y^m)$ is drawn from $\mathcal{D}$ (resp. Q^t). To bound the expectation above, we use the domination property of Q^t, i.e., for each $y^1,\ldots,y^m$,

$$\mathrm{Q}^t(y^1,\ldots,y^m) \geq 2^{-O(d)}\cdot\mathcal{D}(y^1,\ldots,y^m).$$

The domination property holds because the prefix of a random seed to U^t in sampling according to Q^t matches the d-bit description of a sampling algorithm for $\mathcal{D}$ with probability at least $2^{-O(d)}$. Under this event, the conditional distribution of Q^t is statistically identical to $\mathcal{D}$.

To show Lemma 4.3.5, we generalize the above argument to the case of (i) q-query algorithms and (ii) $x^i \neq \varepsilon$. For case (i), we use a fundamental fact on KL divergence for independently and identically selected random variables. In case (ii), we first observe that the expectation in Lemma 4.3.5 is bounded above by

$$\frac{1}{m}\mathop{\mathbb{E}}_{x^1,y^1,\ldots,x^m,y^m\sim\mathcal{D}}\left[\ln\frac{\mathcal{D}(x^1,y^1,\ldots,x^m,y^m|x^1,\ldots,x^m)}{\mathrm{Q}^t(x^1,y^1,\ldots,x^m,y^m|x^1,\ldots,x^m)}\right],$$

where $\mathcal{D}(x^1, y^1, \ldots, x^m, y^m | x^1, \ldots, x^m)$ (resp. $\mathrm{Q}^t(x^1, y^1, \ldots, x^m, y^m | x^1, \ldots, x^m)$) represents the conditional probability that $(x^1, y^1, \ldots, x^m, y^m)$ is drawn from $\mathcal{D}$ (resp. Q^t) given $x^1, \ldots, x^m$. Then, we evaluate the expectation above by observing that the conditional variant of a domination property (given $x^1, \ldots, x^m$) holds in expectation over $x^1, y^1, \ldots, x^m, y^m \sim \mathcal{D}$.

The proof of Theorem 4.3.4 is inspired by the theory of universal prediction [MF98; Hut05], which shows that, informally speaking, minimization of an expected loss under Q^∞ (i.e., a time-*unbounded* universal distribution) yields minimization of an expected loss under computable distributions dominated by Q^∞. We apply their theory to the time-bounded universal distribution Q^t. However, this raises an issue that their theory is established in a statistical setting (i.e., a time-unbounded setting), and it is unclear whether it holds in the time-bounded (complexity theory) regime; e.g., minimizing an expected loss under Q^t can be computationally infeasible. We close this computational-statistical gap by choosing the goal of learners carefully in the formulation of the framework in Section 4.3.3, i.e., we consider the worst-case performance of the cheating learner as the target. In this formulation, we observe that their theory (particularly, the evaluation in [MF98]) holds even in time-bounded settings, which yields Theorem 4.3.4 (see Section 4.9 for the details).

4.3.5 Limitations of Universal Extrapolation

We discuss that several conditions of universal extrapolation are necessary in the sense that relaxing either of the conditions is impossible or harder than solving notorious open questions.

First, universal extrapolation only works in average-case settings, where secret information and samples are selected according to some samplable distributions. This condition is due to the assumption that every efficiently computable function is invertible *on average*, and even partial relaxation of the average-case conditions is hard. For instance, universal extrapolation implies the average-case learnability result discussed in [BFKL93], where the model has two distributions of examples and targets. Improving either of two distributions to the worst-case setting (i.e., all example distributions or all target functions as the original PAC learning model) yields an equivalence result between OWF and a weaker cryptographic primitive called auxiliary-input one-way function [ABX08; Nan21a], which is a long-standing open question that dates back to the work [Ost91]. In fact, [Tre10; Nan21b] showed the necessity of a nonrelativizing proof for such an equivalence result; however, all of the proof techniques in this work are relativizing. Furthermore, improving distributional learning in Corollary 4.2.3 to distributional learning P/poly-samplable distributions (in the worst-case sense) also implies the equivalence between the existence of OWF and the existence of an auxiliary-input one-way function [KMRRSS94; Xia10].

Second, universal extrapolation only works on average over the choice of prediction stages. This condition is not restrictive when samples are selected independently and identically because the prediction is the same at all stages. However, in general cases in which the offline stream has a correlation, a more desirable goal is prediction at an arbitrary stage after sufficiently long observations. We remark that this goal is too much to hope for and *information-theoretically impossible* for the reason mentioned in [NR06], i.e., there is a case in which the distribution on samples can be switched to a completely different distribution at some i-th stage for sufficiently large i. A universal learner has no way to notice the switching at the i-th stage and cannot correctly predict the next outcome.

Third, our universal learner runs in exponential time in the description length d of samplers for selecting secret information to generate offline streams due to Theorem 4.3.2 (do not confuse d with the amount s of secret information). Improving the running time of our learner to polynomial in d

also yields the equivalence between OWF and an auxiliary-input one-way function because such an improved learner succeeds in learning in $\mathsf{poly}(n)$ time even for samplers described by $\mathsf{poly}(n)$-bits, and this is sufficient to break any auxiliary-input one-way function by the known reduction from inverting functions to weak learning [Val84; GGM86; HILL99; ABX08].

Some readers might be skeptical about the novelty of universal learning because the description of samplable distribution is bounded by some constant d, and if the size n of each sample is sufficiently large so that $n \geq 2^d$, then the learner can try all possible distributions in polynomial time (but exponential time in d). Again, note that d is the description length of the distribution that selects $\mathsf{poly}(n)$-bit secret information to generate samples, and the learner does not take exponential time in the amount of the secret information. However, even if we allow exponential time in d (it enables the brute-force of underlying distributions), constructing universal learners based on distribution-specific ones is still challenging in learning ACDs and distributional learning for the following reasons. Applying the distribution-specific learner to all candidates for underlying distributions yields many hypotheses. Although one of them must be a good hypothesis that approximates the target distribution well, it is unclear whether we can identify the good hypothesis efficiently. In general, verification of the goodness of a hypothesis in learning distributions is the problem called Statistical Distance which asks whether two given distributions are statistically close. This problem is known to be SZK-complete [GV99] and efficiently solvable on average if there is no one-way function [OW93] *when query access to both distributions is available* (where the oracle is given *a random seed* and returns *the corresponding sample*). However, in universal learning, query access to the distribution that actually generates samples is unavailable, and it is unclear whether Statistical Distance is solvable in such a case. Furthermore, even in time-unbounded settings, it is not clear whether verification of the goodness of a hypothesis is possible in learning ACDs because the internal state can be updated for each access to a sample. Namely, we cannot even collect samples multiple times for some empirical estimation. By these observations, there seems to be no way to verify the performance of the hypothesis in universal learning ACDs or perhaps in distributional learning. Particularly in learning ACDs, even if the learner knows the secret information (i.e., the initial state), it is still unclear whether the learner can simulate the distribution of the next outcome because the learner does not know the sampling algorithm. Our universal learner overcomes this difficulty in the sense that it does not depend on any verification of candidates for hypotheses. Note that, in the case of agnostic learning (i.e., Theorem 4.2.2), the verification of hypotheses is possible by the standard empirical estimation [HKLW88], and it is easy to transform distribution-specific learners into a universal learner. Therefore, we do not claim the novelty of universality in Theorem 4.2.2 so much, but even a distribution-specific learner for efficient agnostic learning was not known before this work, as far as we know.

4.4 Related Work

Valiant [Val84] observed that the existence of pseudorandom functions (which is equivalent to the existence of one-way functions [GGM86; HILL99]) implies the hardness of learning for $\mathsf{P/poly}$. A line of subsequent work proved the cryptographic hardness of learning for several natural subclasses [KV94a; AK95; Kha93; KS09; DV21]. They mainly discussed opposite directions compared to our work, i.e., from learning to the non-existence of cryptographic primitives.

Several learnability results have been shown under some related algorithmic assumptions. As mentioned in the introduction, Impagliazzo and Levin [IL90] initiated the study on efficient learning

under the non-existence of OWF, and subsequent works [BFKL93; NR06; NY19] investigated the capability of learning in more standard models. Oliveira and Santhanam [OS17] presented the characterization of the existence of exponentially secure one-way functions based on the exponential hardness of PAC learning with membership queries on the uniform example distribution in the *non-uniform* setting, which is incomparable to our result. Nanashima [Nan21a] considered another average-case variant of PAC learning, where a target function is selected according to a fixed samplable distribution but examples are selected according to unknown (possibly not efficiently samplable) distributions as in the original PAC learning model. He showed that PAC leaning is feasible in such a model under the stronger assumption that there is no auxiliary-input one-way function. Although his idea can be applied in the case of infinitely-often one-way functions, the learner only works in the same setting as [BFKL93], which is improved by this work in the same sense as discussed in Section 4.2. A line of study [CIKK16; CIKK17; BCKRS22] presented distribution-specific efficient learners under the stronger assumptions of the existence of natural proofs. Li and Vitányi [LV89] developed a universal PAC learner on simple distributions that contain $\mathsf{P/poly}$-computable distributions under the stronger assumption that every NP problem is easy on average. Hirahara and Nanashima [HN21] also developed a universal agnostic learner that works on every unknown $\mathsf{P/poly}$-samplable distributions over examples under the stronger assumption that every NP problem is easy on average with zero error. Nanashima [Nan20] showed that PAC learning $\mathsf{P/poly}$ is feasible under the non-existence of a cryptographic primitive, named an auxiliary-input local hitting set generator, that has a significantly weaker security condition than OWF. In another line of research, several cryptographic primitives were constructed based on the hardness of learning linear functions with a random noise (e.g. [Reg09; DP12]). In this context, our result can be regarded as a construction of a one-way function whose security is based on much more general assumptions of the average-case hardness of learning.

Our result is also regarded as a generalization of the well-known Yao's next-bit generator theorem [Yao82] in the following sense. The next-bit generator theorem shows that, under the non-existence of OWF, every function that expands secret information (selected uniformly at random) has a next-bit generator that weakly simulates the next-bit of the outcome on average over the choice of the secret information and a position of the simulated bit. Theorem 4.3.3 for the trivial one-query cheating learner provides a universal *next-block* generator theorem, i.e., there exists a universal next-block generator such that for every function that expands secret information (selected according to some samplable distribution) as blocks, the generator strongly simulates the distribution of the next block of the outcome on average over the choice of the secret information and a position of the simulated block. Previously, Vadhan and Zheng [VZ12] gave another related equivalence result between conditional pseudo-entropy and KL-hardness by using the uniform min-max theorem [VZ13], but their work seems to have no direct connection to our work.

Klivans, Lee, and Wan [KLW10] introduced another formulation of agnostic learning on average, where they assume a target function f is selected according to some samplable distribution and ask a learner to learn all target functions that are close to f with high probability over the choice of f. Unfortunately, our learning algorithm does not work under this formulation because every case of adversarial noise must be considered regardless of samplability. Nevertheless, our learner works as long as the unknown adversary that determines noise is selected according to some uniform and efficient sampling mechanism. This computational assumption seems quite natural under the strong form of the Church–Turing thesis.

Several novel techniques for learning natural classes (e.g., decision trees and DNFs) on average

have developed in PAC learning model [JS05; Sel08; Sel09; JLSW11; AC15] and in the agnostic learning model with membership queries [KKMS08; GKK08; KK09; KLW10]. These results are unconditional but work on only some specific distributions, particularly in many cases, the uniform distribution over examples and target functions in the class.

A recent line of work [LP20; LP21c; LP21b; RS21; ACMTV21; IRS22; LP22] characterized the existence of OWF by the average-case meta-complexity, i.e., the average-case hardness of computing minimum description length of a given string. Our result shows that, through the existence of OWF, the feasibility of various learning problems (as in Corollary 4.2.3) is also characterized by the average-case meta-complexity. We further discuss the relationships between [IL90] and meta-complexity in Section 4.6.

Very recently, Hopkins, Kane, Lovett, and Mahajan [HKLM22] established a robust theory of learning in statistical settings, which includes the equivalence between PAC learning and agnostic learning (where they considered the worst-case setting with respect to distributions as in the original models). Their idea rely on unbounded computational resources and seems not to hold when we consider polynomial-time learners. By contrast, we established the robust theory of learning in complexity-theoretic regime but only in average-case settings. These results lead us to expect the robust theory of learning in *computational* and *worst-case* settings. Previously, Xiao [Xia10] showed that distributional learning for P/poly-samplable distributions implies PAC learning under *every* unknown P/poly-samplable distributions over samples by using the technique in [ABX08]; however, the converse remains open.

Organization of the Remainder of This Chapter

The remainder of this chapter is organized as follows. In Section 4.5, we introduce some additional basic notions for our formal arguments. In Section 4.6, we present the full proof of the proposition in [IL90] for universal estimation of probability, whose formal proof was not found in previous work. We also discuss the relationships between [IL90] and meta-complexity in Section 4.6. In Section 4.7, we present our formulation of universal extrapolation and present the formal proof. In Section 4.8, we introduce the general framework for translating universal extrapolation into learning algorithms and present the formal statement and the proof of Theorem 4.3.3. Note that Theorem 4.2.1 is also shown in Section 4.7 as an application. In Section 4.9, we introduce the framework for translating universal extrapolation into learning algorithms for minimizing the expected loss and present the formal statement and the proof of Theorem 4.3.4. Note that Theorem 4.2.2 is also shown in Section 4.9 as an application.

4.5 Additional Preliminaries

For each $t\colon \mathbb{N}\times\mathbb{N}\to\mathbb{N}$, we say that a family $\mathcal{D}=\{\mathcal{D}_{n,z}\}_{n\in\mathbb{N},z\in\{0,1\}^*}$ of distributions (on binary strings) is $t(n,|z|)$-time samplable if there exists a deterministic algorithm D such that, for each $(n,z)\in\mathbb{N}\times\{0,1\}^*$, the distribution of $D(1^n,z,U_{t(n,|z|)})$ is statistically identical to $\mathcal{D}_{n,z}$, and $D(1^n,z,\text{-})$ halts in time $t(n,|z|)$. For every $t(n)$-time samplable distribution $\mathcal{D}=\{\mathcal{D}_n\}_{n\in\mathbb{N}}$, we use the notation $d(\mathcal{D})$ to refer to the description length of the sampler for $\mathcal{D}$ (with respect to the universal Turing machine U), i.e., there exists a $t(n)$-time algorithm D of description length $d(\mathcal{D})$ such that each $\mathcal{D}_n$ is statistically identical to $D(1^n,r)$ for $r\sim\{0,1\}^{t(n)}$. We also use the same notation $d(\mathcal{D})$ for samplable distributions $\mathcal{D}=\{\mathcal{D}_{n,z}\}_{n\in\mathbb{N},z\in\{0,1\}^*}$ indexed by (n,z).

Fact 4.5.1 (Pinsker's inequality)**.** *For every distributions $\mathcal{X}$ and $\mathcal{Y}$,*

$$\mathsf{L}_1(\mathcal{X},\mathcal{Y}) \le \sqrt{\frac{1}{2}\mathrm{KL}(\mathcal{X}||\mathcal{Y})}.$$

Fact 4.5.2. *For every distributions $\mathcal{X}$ and $\mathcal{Y}$ on a discrete domain D and every randomized function f of domain D, if $\mathrm{KL}(\mathcal{X}||\mathcal{Y}) < \infty$, then $\mathrm{KL}(f(\mathcal{X})||f(\mathcal{Y})) \le \mathrm{KL}(\mathcal{X}||\mathcal{Y})$.*

Proof. We verify $\mathrm{KL}(\mathcal{X}||\mathcal{Y}) - \mathrm{KL}(f(\mathcal{X})||f(\mathcal{Y})) \ge 0$ as follows:

$$\begin{aligned}
\mathrm{KL}(\mathcal{X}||\mathcal{Y}) - \mathrm{KL}(f(\mathcal{X})||f(\mathcal{Y})) &= \sum_{x\in D}\mathcal{X}(x)\log\frac{\mathcal{X}(x)}{\mathcal{Y}(x)} - \sum_{a\in\mathrm{Im}f}\Pr[f(\mathcal{X})=a]\log\frac{\Pr[f(\mathcal{X})=a]}{\Pr[f(\mathcal{Y})=a]}\\
&= \sum_{x\in D}\mathcal{X}(x)\log\frac{\mathcal{X}(x)}{\mathcal{Y}(x)} - \sum_{a\in\mathrm{Im}f}\sum_{x\in D}\mathcal{X}(x)\mathbb{1}\{f(x)=a\}\log\frac{\Pr[f(\mathcal{X})=a]}{\Pr[f(\mathcal{Y})=a]}\\
&= \sum_{x\in D}\mathcal{X}(x)\log\frac{\mathcal{X}(x)}{\mathcal{Y}(x)} - \sum_{x\in D}\mathcal{X}(x)\sum_{a\in\mathrm{Im}f}\mathbb{1}\{f(x)=a\}\log\frac{\Pr[f(\mathcal{X})=a]}{\Pr[f(\mathcal{Y})=a]}\\
&= \sum_{x\in D}\mathcal{X}(x)\left(\log\frac{\mathcal{X}(x)}{\mathcal{Y}(x)} - \log\frac{\Pr[f(\mathcal{X})=f(x)]}{\Pr[f(\mathcal{Y})=f(x)]}\right)\\
&= \mathop{\mathbb{E}}_{x\sim\mathcal{X}}\left[\log\frac{\Pr_{\mathcal{X}}[\mathcal{X}=x|f(\mathcal{X})=f(x)]}{\Pr_{\mathcal{Y}}[\mathcal{Y}=x|f(\mathcal{Y})=f(x)]}\right].
\end{aligned}$$

For each $x\in D$, let $\mathcal{X}'_{f(x)}$ (resp. $\mathcal{Y}'_{f(x)}$) be the conditional distribution of $\mathcal{X}$ (resp. $\mathcal{Y}$) given the event that $f(\mathcal{X})=f(x)$ (resp. $f(\mathcal{Y})=f(x)$). By regarding that a sampling $x\sim\mathcal{X}$ is performed as $x'\sim\mathcal{X}$ and $x\sim\mathcal{X}'_{f(x')}$, we have

$$\begin{aligned}
\mathrm{KL}(\mathcal{X}||\mathcal{Y}) - \mathrm{KL}(f(\mathcal{X})||f(\mathcal{Y})) &= \mathop{\mathbb{E}}_{x'\sim\mathcal{X}}\left[\mathop{\mathbb{E}}_{x\sim\mathcal{X}'_{f(x')}}\left[\log\frac{\Pr_{\mathcal{X}}[\mathcal{X}=x|f(\mathcal{X})=f(x)]}{\Pr_{\mathcal{Y}}[\mathcal{Y}=x|f(\mathcal{Y})=f(x)]}\right]\right]\\
&= \mathop{\mathbb{E}}_{x'\sim\mathcal{X}}\left[\mathrm{KL}(\mathcal{X}'_{f(x')}||\mathcal{Y}'_{f(x')})\right]\\
&\ge 0,
\end{aligned}$$

where the inequality follows from the non-negativity of the KL divergence. □

4.5.1 Domination Property

We introduce the important *domination* property of the time-bounded universal distribution Q^t.

Lemma 4.5.3 (domination)**.** *For every distribution $\mathcal{D}$ that has a t_D-time sampler of description length d, there exists $t_0\in\mathbb{N}$ such that for every $x\in\{0,1\}^*$, and for every $t\in\mathbb{N}$ with $t\ge t_0$, it holds that $\mathrm{Q}^t(x)\ge\mathcal{D}(x)/2^{O(d)}$. Furthermore, $t_0=\tau_{\mathrm{dom}}(d,t_D)$ for a universal polynomial τ_{dom} (that depends on U).*

Proof. Let $M\in\{0,1\}^d$ be the binary encoding of the sampler for $\mathcal{D}$. If τ_{dom} is sufficiently large (depending only on U), then any $t\ge\tau_{\mathrm{dom}}(d,t_D)$ is sufficiently large so that (i) U^t simulates M and (ii) $\langle M,x\rangle$ (where x is a seed for M) is encoded by t bits. Therefore, if the prefix of random seed r to $U^t(r)$ corresponds to $\langle M,\text{-}\rangle$, then the conditional distribution of Q^t is statistically equivalent to $\mathcal{D}$. The lemma holds because the condition is satisfied with probability at least $2^{-O(d)}$. □

4.6 Estimating Universal Probability and Kolmogorov Complexity

In this section, we formally prove the following theorem, which was introduced as the intermediate step in [IL90] with only a very high-level proof sketch.

Theorem 4.6.1 ([IL90, "Proposition" 1])**.** *The following are equivalent.*

1. *There exists no infinitely-often one-way function.*
2. *There exists a randomized polynomial-time algorithm M such that, for every $\mathcal{D} \in \mathrm{PSAMP}$, there exists a polynomial t_0 such that for all large $n \in \mathbb{N}$, for every integer $t \geq t_0(n)$, for every $\delta^{-1} \in \mathbb{N}$,*
$$\Pr_{\substack{x\sim\mathcal{D}_n \\ M}}\left[\mathrm{Q}^t(x)\cdot(1-\delta) \leq M(x,1^t,1^{\delta^{-1}}) \leq \mathrm{Q}^t(x)\cdot(1+\delta)\right] \geq 1-\delta.$$

Furthermore, we show that Theorem 4.6.1 yields a universal algorithm that approximates the resource-unbounded Kolmogorov complexity of a string chosen from *unknown* samplable distributions. This result improves the recent result of [IRS22] that constructed an efficient algorithm $M_{\mathcal{D}}$ that approximates the Kolmogorov complexity of x drawn from any fixed samplable distribution.

Theorem 4.6.2. *The following are equivalent.*

1. *There exists no infinitely-often one-way function.*
2. *There exists a randomized polynomial-time algorithm M such that, for every $\mathcal{D} \in \mathrm{PSAMP}$, there exists a polynomial t_0 such that for all large $n \in \mathbb{N}$, for every integer $t \geq t_0(n)$, for every $\delta^{-1} \in \mathbb{N}$,*
$$\Pr_{\substack{x\sim\mathcal{D}_n \\ M}}\left[\mathrm{K}(x) - \log(1/\delta) - O(\log t) \leq M(x,1^t,1^{\delta^{-1}}) \leq \mathrm{K}(x)\right] \geq 1-\delta.$$

The proof of Theorem 4.6.1 relies on another result of [IL89; IL90], which enables us to estimate the probability $\mathcal{D}(x)$ for a string x drawn from a *known* distribution $\mathcal{D}$. We prove this in Section 4.6.1. In Section 4.6.2, we apply this to the time-bounded universal distribution, which yields a proof of Theorem 4.6.1. Finally, we complete a proof of Theorem 4.6.2 in Section 4.6.3.

The proof of Theorem 4.6.1 ($1 \Rightarrow 2$) also yields the same theorem for $\mathrm{Q}^{t,*}(x)$ instead of $\mathrm{Q}^t(x)$ by replacing the universal Turing machine U^t with the truncated universal Turing machine that outputs the prefix of U^t. Remember that $\mathrm{Q}^{t,*}(x)$ is the probability that the prefix of a sample $y \sim \mathrm{Q}^t$ matches x. We will use the universal approximation for $\mathrm{Q}^{t,*}$ in one of the proofs of the universal extrapolation theorem in Section 4.7.

Theorem 4.6.3 ([IL90, "Proposition" 1] for $\mathrm{Q}^{t,*}$)**.** *If there is no infinitely-often one-way function, then there exist a randomized algorithm M and a polynomial τ_M such that for every $t_D(n)$-samplable distribution $\mathcal{D} = \{\mathcal{D}_n\}_{n\in\mathbb{N}}$, there exists $n_0 \in \mathbb{N}$ such that for every $n, t, \delta^{-1} \in \mathbb{N}$ with $n \geq n_0$ and $t \geq \tau_M(t_D(n))$,*
$$\Pr_{x\sim\mathcal{D}_n,M}\left[(1-\delta)\mathrm{Q}^{t,*}(x) \leq M(x;1^t,1^{\delta^{-1}}) \leq (1+\delta)\mathrm{Q}^{t,*}(x)\right] \geq 1-\delta,$$
where $n_0 = n'(d(\mathcal{D}))$ for a universal function $n'\colon \mathbb{N} \to \mathbb{N}$.

For convenience, we introduce a notation q^t as follows.

Definition 4.6.4 (implicit in [IL90])**.** *For every $t \in \mathbb{N}$ and every string $x \in \{0,1\}^*$, we define*

$$\mathrm{q}^t(x) := -\log \mathrm{Q}^t(x) = -\log \Pr_{d \sim \{0,1\}^t}\left[U^t(d) = x\right].$$

(If $\mathrm{Q}^t(x) = 0$, then we regard $\mathrm{q}^t(x)$ as ∞.)

The value $\mathrm{q}^t(x)$ can be regarded as randomized time-bounded Kolmogorov complexity; in fact, we can observe the equivalence between q^t and the time-bounded probabilistic Kolmogorov complexity pK^t, recently studied in [GKLO22], up to an additive logarithmic factor and a polynomial overhead of the time-bound (see Section 4.6.4). In the resource-unbounded case, it is known that $\lim_{t\to\infty} \mathrm{q}^t(x)$ is equal to the Kolmogorov complexity of x up to an additive constant [LV19].

We also observe the upper bound on q^t, which follows from the domination property.

Proposition 4.6.5 (Implicit in [IL90])**.** *For every $\mathcal{D} \in \mathrm{PSAMP}$, there exists a polynomial t_0 such that for every polynomial $t \geq t_0$ and for every $n \in \mathbb{N}$ and every $x \in \mathrm{supp}(\mathcal{D}_n)$,*

$$\mathrm{q}^{t(n)}(x) \leq -\log \mathcal{D}_n(x) + \log t(n).$$

Proof. Let S be a randomized polynomial-time algorithm that, on input 1^n, outputs a string distributed according to $\mathcal{D}_n$. Since $S(1^n; \text{-})$ is described by $O(\log n)$ bits, by Lemma 4.5.3, there exist polynomials $t_0(n)$ and $p(n)$ such that, for every $n \in \mathbb{N}$, every $t \geq t_0(n)$, and every $x \in \mathrm{supp}(\mathcal{D}_n)$,

$$2^{-\mathrm{q}^t(x)} = \mathrm{Q}^t(x) \geq \frac{1}{p(n)} \cdot \mathcal{D}_n(x).$$

□

4.6.1 Estimating the Probability with respect to Known Distributions

We first apply a standard hardness amplification technique to obtain an inverter that takes an additional parameter $\delta^{-1} \in \mathbb{N}$ and successfully inverts a one-way function with probability $1 - \delta$.

Proposition 4.6.6. *If there exists no infinitely-often one-way function, then for every polynomial-time-computable family $f = \left\{f_n \colon \{0,1\}^{s(n)} \to \{0,1\}^{t(n)}\right\}_{n\in\mathbb{N}}$, there exists a randomized polynomial-time algorithm A such that for every $n \in \mathbb{N}$ and every $\delta^{-1} \in \mathbb{N}$,*

$$\Pr_{x \sim \{0,1\}^{s(n)}, A}[A(f_n(x); 1^n, 1^{\delta^{-1}}) \notin f_n^{-1}(f_n(x))] \leq \delta.$$

Proof Sketch. We define a new family $f' = \left\{f_{\langle n,k\rangle}\right\}_{\langle n,k\rangle\in\mathbb{N}}$ so that

$$f'_{\langle n,k\rangle}(x_1, \ldots, x_k) = (f(x_1), \ldots, f(x_k))$$

for every $x_1, \ldots, x_k \in \{0,1\}^{s(n)}$, where $\langle \text{-}, \text{-}\rangle \colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ is a bijection. Yao's amplification theorem [Yao82; Gol01] shows that inverting f_n with probability $1 - \delta$ reduces to the task of inverting $f_{\langle n,k\rangle}$ for some $k = \mathsf{poly}(n, \delta^{-1})$. □

We now show that there exists an algorithm that approximates $\mathcal{D}_n(x)$ for a string x drawn from $\mathcal{D}_n$.

Lemma 4.6.7 ([IL89; IL90; Imp]). *Assume that there exists no infinitely-often one-way function. Then, for every $\mathcal{D} \in \mathrm{PSAMP}$, there exists a randomized polynomial-time algorithm A such that for all large $n \in \mathbb{N}$ and all large $\delta^{-1} \in \mathbb{N}$,*

$$\Pr_{\substack{x \sim \mathcal{D}_n \\ A}}\left[(1-\delta)\cdot \mathcal{D}_n(x) \le A(x, 1^n, 1^{\delta^{-1}}) \le (1+\delta)\cdot \mathcal{D}_n(x)\right] \ge 1-\delta.$$

Although a proof of this result appears to be known to researchers in the 1990s (e.g., [OW93]), we are not aware of any published proof. In fact, recent papers [Nan21b; IRS21] provide a weaker algorithm that approximates $\mathcal{D}_n(x)$ within some constant factor (instead of a $(1+\delta)$-factor for an arbitrary small $\delta > 0$), which is further used in [LP21a]. Although such a weak algorithm is sufficient for most applications in the previous studies, it is insufficient for our purpose of showing the universal extrapolation theorem. To the best of our knowledge, the following is the first written proof of Lemma 4.6.7. The proof closely follows the proof of the construction of a distributional inverter presented in the PhD thesis of Impagliazzo [Imp].

Proof of Lemma 4.6.7. Let f be the sampler of $\mathcal{D}$. That is, $f = \{f_n : \{0,1\}^n \to \{0,1\}^n\}_{n\in\mathbb{N}}$ and the distribution of $f_n(x)$ over $x \sim \{0,1\}^n$ is identical to $\mathcal{D}_n$. (Without loss of generality, we assume that the input length of f_n and the output length are identical.)

Let $h_\ell \colon \{0,1\}^n \to \{0,1\}^\ell$ be a pairwise independent hash. Consider the family $g = \left\{g_{\langle n,\ell\rangle}\right\}_{n,\ell\in\mathbb{N}}$ of functions defined as follows. $g_{\langle n,\ell\rangle}(x, h_\ell) := (f_n(x), h_\ell(x), h_\ell)$. Here, we identify h_ℓ with the random bits used to generate h_ℓ. For simplicity, in what follows, we omit the subscripts of g because n and ℓ are clear from the context.

Since g is computable in polynomial time, the assumption implies that there exists a randomized polynomial-time algorithm I such that

$$\Pr_{x,h_\ell,I}\left[I(g(x,h_\ell); 1^{\langle n,\ell\rangle}, 1^{\epsilon^{-1}}) \in g^{-1}(g(x,h_\ell))\right] \ge 1-\epsilon. \tag{1}$$

Here, the probability is taken over all x, h_ℓ and the internal randomness of I. For notational simplicity, we omit I from the subscript of probabilities in what follows. We also often omit the auxiliary input $(1^{\langle n,\ell\rangle}, 1^{\epsilon^{-1}})$ from the input to I.

Using this inverter I, we present the definition of the algorithm A: The algorithm A takes $(y, 1^n, 1^{\delta^{-1}})$ as input and sets $\epsilon := 1/\mathsf{poly}(n/\delta)$ for some polynomial poly to be chosen later. Then, for each $\ell = n + \log(1/\delta), \dots, 0$ in decreasing order, A estimates

$$v_\ell := \Pr_{\substack{r\sim\{0,1\}^\ell \\ h_\ell}}\left[I(y,r,h; 1^{\langle n,\ell\rangle}, 1^{\epsilon^{-1}}) \in g^{-1}(y,r,h)\right]$$

by randomly sampling r and h several times. Let $\widetilde{v_\ell}$ be the estimate of v_ℓ computed by A. If $\widetilde{v_\ell} \le \delta$, the algorithm A continues to the next loop with ℓ decreased by 1. If $\widetilde{v_\ell} > \delta$, the algorithm A outputs $\widetilde{v_\ell} \cdot 2^{\ell-n}$ and halts.

Below, we prove that the output of A approximates $\mathcal{D}_n(y) = 2^{-n}\cdot\left|f_n^{-1}(y)\right|$ with probability at least $1-\delta$.

By Hoeffding's inequality, the algorithm A can compute $\widetilde{v_\ell}$ such that $|\widetilde{v_\ell} - v_\ell| \le \epsilon$ in time $\mathsf{poly}(n/\epsilon)$ with probability at least $1-2^{-n}$ over the internal randomness of A. In what follows, we may assume that $\widetilde{v_\ell}$ satisfies $|\widetilde{v_\ell} - v_\ell| \le \epsilon$. We may also assume that $\delta \ge 2^{-n}$, as otherwise a brute-force search can be used to compute $\left|f_n^{-1}(y)\right|$ in time $2^{O(n)}$.

Fix an input $y \in \text{image}(f_n)$. Let $X \subseteq \{0,1\}^n$ denote $f_n^{-1}(y)$. Let $\ell \in \mathbb{N}$ be the last integer in the algorithm A on input y.

Claim 4.6.8. *Assume that A halts with ℓ. Then,*

$$\widetilde{v_\ell} - \epsilon \leq |X| \cdot 2^{-\ell}.$$

In particular,

$$\delta - \epsilon \leq |X| \cdot 2^{-\ell}.$$

Proof. Since $|\widetilde{v_\ell} - v_\ell| \leq \epsilon$, it suffices to show $v_\ell \leq |X| \cdot 2^{-\ell}$. Observe that I succeeds to invert g on (y, r, h_ℓ) only if (y, r, h_ℓ) is in the image of g, in which case $r = h_\ell(x)$ for some $x \in f^{-1}(y) = X$. Since $r \sim \{0,1\}^\ell$, by a union bound, the probability v_ℓ that I succeeds to invert g on (y, r, h_ℓ) is at most $|X| \cdot 2^{-\ell}$. The "in particular" part follows from the fact that $\widetilde{v_\ell} > \delta$ when A halts. ◇

Claim 4.6.9. *With probability at least $1 - \sqrt{\epsilon} \cdot 2n$ over a choice of $y := f_n(x)$ and $x \sim \{0,1\}^n$, it holds that for every ℓ,*

$$|X| \cdot 2^{-\ell} \cdot \left(1 - |X| \cdot 2^{-\ell}\right) \cdot \left(1 - \sqrt{\epsilon}\right) \leq \widetilde{v_\ell} + \epsilon.$$

Proof. For notational simplicity, we omit the subscript ℓ of h_ℓ. Fix $y \in \text{image}(f_n)$. For a hash function h, let S_h denote the set of all the strings $x \in X$ such that $h(x) \neq h(x')$ for every $x' \in X \setminus \{x\}$. Let $h(S_h)$ denote the image of S_h under h. For every $x \in X$, by a union bound, we have $\Pr_h[x \notin S_h] \leq |X| \cdot 2^{-\ell}$. In particular, we obtain

$$\mathop{\mathbb{E}}_h[|S_h|] \geq |X| \cdot (1 - |X| \cdot 2^{-\ell}). \tag{2}$$

Let $h(S_h)$ denote the image of S_h under h. Under the event that $r \in h(S_h)$, the random variable (r, h) is identical to the distribution of $(h(x), h)$ over $x \sim S_h$. Thus, we obtain

$$\begin{aligned} v_\ell &= \Pr_{r,h}\left[I(y, r, h; 1^{\langle n,\ell\rangle}, 1^{\epsilon^{-1}}) \in g^{-1}(y, r, h)\right] \\ &\geq \Pr[r \in h(S_h)] \cdot \Pr_{\substack{h \\ x \sim S_h}}\left[I(f(x), h(x), h) \in g^{-1}(f(x), h(x), h)\right] \end{aligned}$$

For the first factor, we have

$$\Pr[r \in h(S_h)] = 2^{-\ell} \cdot \mathop{\mathbb{E}}_h[|h(S_h)|] \geq 2^{-\ell} \cdot |X| \cdot (1 - |X| \cdot 2^{-\ell}),$$

where the last inequality follows from Eq. (2). The second factor can be bounded from below by

$$\begin{aligned} &\Pr_{\substack{h \\ x \sim X}}\left[I(f(x), h(x), h) \in g^{-1}(f(x), h(x), h)\right] \\ = &\Pr_{\substack{h \\ x \sim \{0,1\}^n}}\left[I(f(x), h(x), h) \in g^{-1}(f(x), h(x), h) \;\middle|\; y = f(x)\right] \end{aligned}$$

because $S_h \subseteq X$.

Finally, we consider the random variable y distributed according to $f_n(x)$ over $x \sim \{0,1\}^n$. By Markov's inequality and Eq. (1), with probability at least $1 - \sqrt{\epsilon}$ over $y \sim f_n(U_n)$, it holds that

$$\Pr_{\substack{h \\ x \sim \{0,1\}^n}}\left[I(f(x), h(x), h) \in g^{-1}(f(x), h(x), h) \;\middle|\; y = f(x)\right] \geq 1 - \sqrt{\epsilon}.$$

By taking a union bound over all $\ell \le n + \log(1/\delta) \le 2n$, with probability at least $1 - \sqrt{\epsilon} \cdot 2n$ over $y \sim f_n(U_n)$, the same event happens, in which case we have

$$v \ge 2^{-\ell} \cdot |X| \cdot (1 - |X| \cdot 2^{-\ell}) \cdot (1 - \sqrt{\epsilon}).$$

The claim follows by $v_\ell \le \widetilde{v_\ell} + \epsilon$. ◇

Let ℓ be the last integer in the algorithm A on input y. Since A did not halt in all the preceding loops, we have $\widetilde{v}_{\ell'} \le \delta$ for every $\ell' > \ell$. By Claim 4.6.9, we obtain

$$|X| \cdot 2^{-\ell'} \cdot \left(1 - |X| \cdot 2^{-\ell'}\right) \cdot \left(1 - \sqrt{\epsilon}\right) \le \widetilde{v}_{\ell'} + \epsilon \le \delta + \epsilon.$$

Let $\gamma := |X| \cdot 2^{-\ell}$. Then, we have

$$2^{-k}\gamma \cdot (1 - 2^{-k}\gamma) \cdot (1 - \sqrt{\epsilon}) \le \delta + \epsilon$$

for every integer $k \ge 1$. In particular, we obtain $\gamma \le 4\delta$ for sufficiently small δ and ϵ. (Otherwise, one can choose k so that $2^{-k}\gamma \in [2\delta, 4\delta]$, which is a contradiction.) We also have $\delta - \epsilon < \gamma$ from Claim 4.6.8. We choose a small ϵ so that $\epsilon \ll \delta^2$. Then, $\sqrt{\epsilon} \ll \delta$ and $\epsilon \ll \gamma\delta$.

By Claim 4.6.9 and $\gamma \le 4\delta$, we obtain

$$\gamma \cdot (1 - O(\delta)) \le \gamma \cdot (1 - 4\delta) \cdot (1 - \sqrt{\epsilon}) - \epsilon \le \widetilde{v}_\ell.$$

By Claim 4.6.8, we have

$$\widetilde{v}_\ell \le \gamma + \epsilon \le \gamma \cdot (1 + \delta).$$

Recall that the output of A is $\widetilde{v}_\ell \cdot 2^{\ell - n}$. We conclude that

$$|X| \cdot 2^{-n} \cdot (1 - O(\delta)) \le \widetilde{v}_\ell \cdot 2^{\ell - n} \le |X| \cdot 2^{-n} \cdot (1 + \delta).$$

□

4.6.2 Universal Approximation of $\mathbf{Q^t}$ and $\mathbf{Q^{t,*}}$

We now apply Lemma 4.6.7 to the time-bounded universal distribution and obtain the "first written" proof of the theorem of Impagliazzo and Levin [IL90].

Proof of Theorem 4.6.1. We only prove Item 1 ⇒ Item 2. Consider the time-bounded universal distribution $\mathcal{M} = \{\mathcal{M}_t\}_{t \in \mathbb{N}}$ defined as follows: $\mathcal{M}_t$ is the distribution of $U^t(d)$ over a random choice of $d \sim \{0,1\}^t$. Observe $\mathcal{M} \in \mathrm{PSAMP}$. Applying Lemma 4.6.7 to $\mathcal{M}$, we obtain a randomized polynomial-time algorithm A such that for all $t \in \mathbb{N}$ and all $\delta^{-1} \in \mathbb{N}$,

$$\Pr_{\substack{x \sim \mathcal{M}_t \\ A}}\left[(1-\delta) \cdot \mathcal{M}_t(x) \le A(x, 1^t, 1^{\delta^{-1}}) \le (1+\delta) \cdot \mathcal{M}_t(x)\right] \ge 1 - \delta.$$

Observe that $\mathcal{M}_t(x) = \mathrm{Q}^t(x)$. Thus, it remains to show that for every $\mathcal{D} \in \mathrm{PSAMP}$, the algorithm A succeeds on $\mathcal{D}$. To show this, we use the argument of domination. Fix $\mathcal{D} \in \mathrm{PSAMP}$. By Proposition 4.6.5, there exists a polynomial t_0 such that $\mathcal{D}_n(x) \le \mathcal{M}_t(x) \cdot t$ for every $n \in \mathbb{N}$,

every $x \in \mathrm{supp}(\mathcal{D}_n)$, and every $t \geq t_0(n)$. Let $E(x, A)$ denote the event that $A(x, 1^t, 1^{\delta^{-1}}) \notin [(1 \pm \delta) \cdot \mathcal{M}_t(x)]$. Then, for any $n \in \mathbb{N}$ and any $t \geq t_0(n)$, we have

$$\begin{aligned}
\Pr_{X \sim \mathcal{D}_n, A}[E(X, A)] &= \sum_{x \in \mathrm{supp}(\mathcal{D}_n)} \Pr_A[E(x, A)] \cdot \mathcal{D}_n(x) \\
&\leq \sum_{x \in \mathrm{supp}(\mathcal{M}_t)} \Pr_A[E(x, A)] \cdot \mathcal{M}_t(x) \cdot t \\
&= \Pr_{X \sim \mathcal{M}_t}[E(X, A)] \cdot t, \\
&\leq \delta \cdot t.
\end{aligned}$$

We now define an algorithm M so that $M(x, 1^t, 1^{\delta^{-1}}) := A(x, 1^t, 1^{(t\delta)^{-1}})$. Then, we obtain

$$\Pr_{\substack{x \sim \mathcal{D}_n \\ M}}\left[M(x, 1^t, 1^{t \cdot \delta^{-1}}) \notin \left[\mathrm{Q}^t(x) \cdot (1 \pm \delta)\right]\right] \leq (t \cdot \delta^{-1})^{-1} \cdot t \leq \delta.$$

□

We can also show Theorem 4.6.3 by replacing $\mathcal{M} = \{\mathcal{M}_t\}_{t \in \mathbb{N}}$ in the proof above with $\mathcal{M} = \{\mathcal{M}_{\langle t,i \rangle}\}_{t,i \in \mathbb{N}}$, where each $\mathcal{M}_{\langle t,i \rangle}$ is the distribution of $U^t(d)_{[i]}$ over $d \sim \{0,1\}^t$, and defining M as $M(x; 1^t, 1^{\delta^{-1}}) := A(x, 1^{\langle t, |x| \rangle}, 1^{(t\delta)^{-1}})$.

4.6.3 Universal Approximation of Kolmogorov Complexity

We now use Theorem 4.6.1 to estimate the resource-unbounded Kolmogorov complexity of a string drawn from any unknown distribution. To this end, we use the fact that no efficient algorithm can produce strings with high computational depth, i.e., $\mathrm{q}^t(x) \approx \mathrm{K}(x)$ for most strings x produced by efficient algorithms [AFMV06; AF09; Hir21b].

Lemma 4.6.10. *For any $\mathcal{D} \in \mathrm{PSAMP}$, there exists a polynomial t_0 such that for every $n \in \mathbb{N}$ and every polynomial $t \geq t_0$,*

$$\Pr_{x \sim \mathcal{D}_n}\left[\mathrm{q}^{t(n)}(x) - \mathrm{K}(x) > k\right] \leq 2^{-k + O(\log t(n))}.$$

Proof. By Proposition 4.6.5, there exists a polynomial t_0 such that for every polynomial $t \geq t_0$, every $n \in \mathbb{N}$, and every $x \in \mathrm{supp}(\mathcal{D}_n)$,

$$2^{\mathrm{q}^{t(n)}(x) - \log t(n)} \leq \frac{1}{\mathcal{D}_n(x)}.$$

In addition, we can assume that $\mathcal{D}_n$ is sampled in time $t(n)$, and $|x| \leq t(n)$ for every $x \in \mathrm{supp}(\mathcal{D}_n)$.

For every polynomial $t \geq t_0$ and every $n \in \mathbb{N}$,

$$\begin{aligned}
\mathop{\mathbb{E}}_{x\sim\mathcal{D}_n}\left[2^{-\mathrm{K}(x)+\mathrm{q}^{t(n)}(x)-\log t(n)}\right] \leq \mathop{\mathbb{E}}_{x\sim\mathcal{D}_n}\left[\frac{2^{-\mathrm{K}(x)}}{\mathcal{D}_n(x)}\right] &= \sum_{x\in\mathrm{supp}(\mathcal{D}_n)} \mathcal{D}_n(x)\frac{2^{-\mathrm{K}(x)}}{\mathcal{D}_n(x)} \\
&= \sum_{x\in\mathrm{supp}(\mathcal{D}_n)} 2^{-\mathrm{K}(x)} \\
&\leq \sum_{i=1}^{2t(n)} \sum_{\substack{x\in\{0,1\}^*:\\ |x|\leq t(n),\mathrm{K}(x)=i}} 2^{-i} \\
&\leq \sum_{i=1}^{2t(n)} 2^i \cdot 2^{-i} = 2t(n),
\end{aligned}$$

This implies the lemma as follows:

$$\begin{aligned}
\Pr_{x\sim\mathcal{D}_n}[\mathrm{q}^{t(n)}(x) - \mathrm{K}(x) > k] &= \Pr_{x\sim\mathcal{D}_n}[2^{\mathrm{q}^{t(n)}(x)-\mathrm{K}(x)-\log t(n)} > 2^{k-\log t(n)}] \\
&\leq \frac{\mathbb{E}_{x\sim\mathcal{D}_n}\left[2^{-\mathrm{K}(x)+\mathrm{q}^{t(n)}(x)-\log t(n)}\right]}{2^{k-\log t(n)}} \\
&\leq 2^{-k+O(\log t(n))},
\end{aligned}$$

where the first inequality follows from Markov's inequality. □

We also note that $\mathrm{K}(x)$ is a lower bound for $\mathrm{q}^t(x)$.

Fact 4.6.11 (e.g., in [LV19, Chapter 4])**.** *For every $x \in \{0,1\}^*$ and every $t \in \mathbb{N}$,*

$$\mathrm{K}(x) \leq \mathrm{q}^t(x) + O(\log t).$$

With the ingredients developed so far, we now prove the main result of this section.

Proof of Theorem 4.6.2. We only prove Item 1 ⇒ Item 2, as the converse can be easily proved using [HILL99] (see [IRS22]). Using Theorem 4.6.1, let M be the algorithm of Item 2. We define an algorithm M' so that $M'(x, 1^t, 1^{\delta^{-1}}) := -\log M(x, 1^t, 1^{\delta^{-1}})$. By the property of M, with probability at least $1-\delta$ over $x \sim \mathcal{D}_n$ and the internal randomness of M, it holds that

$$\mathrm{q}^t(x) - 1 \leq -\log M(x, 1^t, 1^{\delta^{-1}}) \leq \mathrm{q}^t(x) + 1.$$

By Lemma 4.6.10 and Fact 4.6.11, with probability at least $1-\delta$ over $x \sim \mathcal{D}_n$, it holds that

$$\mathrm{K}(x) \leq \mathrm{q}^t(x) + O(\log t) \leq \mathrm{K}(x) + O(\log t) + \log(1/\delta).$$

By a union bound, with probability at least $1-2\delta$ over $x \sim \mathcal{D}_n$, it holds that

$$\mathrm{K}(x) - O(\log t) \leq M'(x, 1^t, 1^{\delta^{-1}}) \leq \mathrm{K}(x) + O(\log t) + \log(1/\delta).$$

The claim follows by subtracting $O(\log t)+\log(1/\delta)$ from the output of M' and choosing a sufficiently large polynomial t_0. □

4.6.4 Universal Distribution and Probabilistic Kolmogorov Complexity

In this section, we show that q^t (in Definition 4.6.4) is equivalent to the time-bounded probabilistic Kolmogorov complexity pK^t up to an additive logarithmic factor and a polynomial overhead of the time-bound. Note that the latter notion is recently studied by Goldberg, Kabanets, Lu, and Oliveira [GKLO22] in the context of meta-complexity. Further background on probabilistic Kolmogorov complexity can also be found in [LO22].

For future work, we consider a general case in which an auxiliary advice string is given. First, we extend the definition of q^t to such a case.

Definition 4.6.12 (implicit in [IL90])**.** *For every $t \in \mathbb{N}$ and every $z \in \{0,1\}^*$, we define the t-time-bounded universal distribution Q^t_z given z as the distribution of $U^t(r;z)$ for $r \sim \{0,1\}^t$, where the universal Turing machine U is given query access to each bit of z.*

For every $t \in \mathbb{N}$ and every $x, z \in \{0,1\}^*$, we define $\mathrm{q}^t(x|z)$ as

$$\mathrm{q}^t(x|z) = -\log \mathrm{Q}^t_z(x).$$

(If $\mathrm{Q}^t_z(x) = 0$, then we regard $\mathrm{q}^t(x|z)$ as ∞.)

Note that $\mathrm{q}^t(x|\varepsilon)$ is equal to $\mathrm{q}^t(x)$ in Definition 4.6.4.

The time-bounded probabilistic Kolmogorov complexity pK^t is defined as follows.

Definition 4.6.13 (Probabilistic Kolmogorov complexity [GKLO22])**.** *For every $t \in \mathbb{N}$ and every $x, z \in \{0,1\}^*$, we define the t-time-bounded Kolmogorov complexity of x given z as*

$$\mathrm{pK}^t(x|z) = \min\left\{k \in \mathbb{N} : \Pr_{r\sim\{0,1\}^t}\left[\exists \pi \in \{0,1\}^k \text{ s.t. } U^t(\pi, r; z) = x\right] \geq 2/3\right\},$$

where the universal Turing machine U is given query access to each bit of z. (If there is no such $p \in \{0,1\}^*$, then we regard $\mathrm{pK}^t(x|z)$ as ∞ for convenience.)

Below, we only consider the case in which $\mathrm{q}^t(x|z) < \infty$ and $\mathrm{pK}^t(x|z) < \infty$.

The equivalence between q^t and pK^t is stated as follows. Note that the second statement follows from the optimal coding theorem for pK^t proved by Lu, Oliveira, and Zimand [LOZ22].

Proposition 4.6.14. *For every $t \in \mathbb{N}$ and every $x, z \in \{0,1\}^*$,*

$$\mathrm{q}^{O(t)}(x|z) \leq \mathrm{pK}^t(x|z) + O(\log t).$$

Theorem 4.6.15 ([LOZ22])**.** *For every $t \in \mathbb{N}$ and every $x, z \in \{0,1\}^*$,*

$$\mathrm{pK}^{p(t)}(x|z) \leq \mathrm{q}^t(x|z) + \log p(t),$$

where p is a universal polynomial that depends on only U.

Proof of Proposition 4.6.14. By the definition of $\mathrm{pK}^t(x|z)$, there exist at least $(2/3)\cdot 2^t$ random seeds $r \in \{0,1\}^t$ that have a program $\pi_r \in \{0,1\}^{\mathrm{pK}^t(x|z)}$ such that $U^t(\pi_r, r; z) = x$. Without loss of generality, we can assume that $\mathrm{pK}^t(x|z) \leq t$. By selecting large enough $t' = O(t)$, the probability that the prefix of a random seed $r' \sim \{0,1\}^{t'}$ corresponds to the program $\langle \pi_r, r\rangle$ is at least $2^{-\mathrm{pK}^t(x|z)-O(\log t)-t}$ (by the standard encoding). Therefore, we obtain that $\mathrm{Q}^{t'}_z(x) \geq (2/3)\cdot 2^t \cdot 2^{-\mathrm{pK}^t(x|z)-O(\log t)-t} = 2^{-\mathrm{pK}^t(x|z)-O(\log t)}$ and $\mathrm{q}^{t'}(x|z) = -\log \mathrm{Q}^{t'}_z(x) \leq \mathrm{pK}^t(x|z) + O(\log t)$. □

Theorem 4.6.15 holds by observing that the proof of the optimal coding theorem for pK^t in [LOZ22, Theorem 5] holds even with additional access to z.

4.7 Universal Extrapolation

In this section, we formulate universal extrapolation and present the formal proof.

Theorem 4.7.1 (Universal Extrapolation)**.** *If there exists no infinitely-often one-way function, then there exist a randomized polynomial-time algorithm* UE *and a polynomial* τ_{UE} *such that for every* $t_D(n)$*-time samplable distribution* $\mathcal{D} = \{\mathcal{D}_n\}_{n\in\mathbb{N}}$ *over binary strings, there exists* $n_0 \in \mathbb{N}$ *such that for all* $n, i, k, t, \epsilon^{-1}, \delta^{-1} \in \mathbb{N}$ *with* $n \geq n_0$ *and* $t \geq \tau_{\mathsf{UE}}(n, t_D(n))$*,*

$$\Pr_{x\sim\mathcal{D}_n}\left[\mathsf{L}_1\left(\mathsf{UE}(x_{[i]}; 1^{\langle n,k,t,\epsilon^{-1},\delta^{-1}\rangle}), \mathsf{Next}_k(\mathrm{Q}^t, x_{[i]})\right) \leq \epsilon\right] \geq 1-\delta,$$

where $n_0 = n_{\mathsf{UE}}(d(\mathcal{D}))$ *for some universal function* $n_{\mathsf{UE}}\colon \mathbb{N}\to\mathbb{N}$ *with* $n_{\mathsf{UE}}(d) = 2^{O(d)}$*.*

We present two proofs of Theorem 4.7.1 by different constructions of UE. The first one is based on a distributional inverter [IL89]. The second one is based on the approximation of the universal a priori probability in Section 4.6, which seems to be the original intention [IL90].

4.7.1 Proof by Distributional Inverter

We present the proof of the universal extrapolation theorem based on distributional inverters. In the proof, we use the following theorem, which is well-known as the equivalence between the existence of one-way functions and the existence of distributional one-way functions.

Theorem 4.7.2 ([IL89; Imp])**.** *The following are equivalent:*

1. *There exists no infinitely-often one-way function;*
2. *For every polynomial-time-computable family* $f = \{f_n\colon \{0,1\}^{s(n)} \to \{0,1\}^{t(n)}\}_{n\in\mathbb{N}}$*, there exists a randomized polynomial-time algorithm* A *such that for every* $n, \epsilon^{-1}, \delta^{-1} \in \mathbb{N}$*,*

$$\Pr_{y\sim f_n(U_{s(n)})}\left[\mathsf{L}_1\left(A(y; 1^n, 1^{\epsilon^{-1}}, 1^{\delta^{-1}}), \mathsf{UnifInv}_{f_n}(y)\right) \leq \epsilon\right] \geq 1-\delta,$$

   where $\mathsf{UnifInv}_f(y)$ *is a random variable selected according to the uniform distribution over* $f^{-1}(y)$ *for each* $y \in \mathrm{Im} f$*.*

Theorem 4.7.2 and the domination property (Lemma 4.5.3) imply distributional inverting according to the universal distribution for a given instance drawn from any unknown samplable distribution.

Lemma 4.7.3. *If there exists no infinitely-often one-way function, then there exist a randomized polynomial-time algorithm* A' *and a polynomial* τ *such that for every* $t_D(n)$*-time samplable distribution* $\mathcal{D} = \{\mathcal{D}_n\}_{n\in\mathbb{N}}$ *over binary strings, there exists* $n_0 \in \mathbb{N}$ *such that for all* $n, i, t, \epsilon^{-1}, \delta^{-1} \in \mathbb{N}$ *with* $n \geq n_0$ *and* $t \geq \tau(n, t_D(n))$*,*

$$\Pr_{x\sim\mathcal{D}_n}\left[\mathsf{L}_1\left(A'(x_{[i]}; 1^{\langle n,t,\epsilon^{-1},\delta^{-1}\rangle}), \mathsf{UnifInv}_{U^t_{[i]}}(x_{[i]})\right) \leq \epsilon\right] \geq 1-\delta,$$

where $U^t_{[i]}$ *denotes the universal Turing machine whose output is truncated to the first* i *bits, i.e.,* $U^t_{[i]}(s) = U^t(s)_{[i]}$ *for each* $s \in \{0,1\}^t$*, and* $n_0 = n'(d(\mathcal{D}))$ *for some universal function* $n'\colon \mathbb{N}\to\mathbb{N}$ *with* $n'(d) = 2^{O(d)}$*.*

Proof. Let A be the randomized algorithm obtained from Theorem 4.7.2 (1⇒2) for a polynomial-time-computable family f defined as, for every $i, t \in \mathbb{N}$ and $s \in \{0,1\}^t$, $f_{\langle i,t\rangle}(s) = U^t_{[i]}(s)$.

For every input, A' is defined as

$$A'(x; 1^{\langle n,t,\epsilon^{-1},\delta^{-1}\rangle}) = A(x; 1^{\langle i,t\rangle}, 1^{\epsilon^{-1}}, 1^{n^{c+1}\delta^{-1}}),$$

where $i = |x|$, and c is a universal constant specified later.

Let $\mathcal{D}' = \{D'_{n,i}\}_{n,i\in\mathbb{N}}$ be a distribution family, where each $\mathcal{D}'_{n,i}$ is a distribution of $x_{[i]}$ for $x \sim \mathcal{D}_n$. Then, $\mathcal{D}'$ is $O(t_D(n))$-time samplable, and $d(\mathcal{D}') = O(d(\mathcal{D}))$.

By Lemma 4.5.3, there exists a polynomial τ_{dom} such that for every $n, i \in \mathbb{N}$ with $i \le t_D(n)$, every $t \ge \tau_{\mathrm{dom}}(d(\mathcal{D}'), \log n, t_D(n))$, and every $x \in \{0,1\}^*$, it holds that $\mathrm{Q}^t(x) \ge \mathcal{D}'_{n,i}(x)/(2^{c\cdot d(\mathcal{D})} \cdot n^c)$ by choosing sufficiently large $c \ge 1$. We can also assume that $d(\mathcal{D}') = O(d(\mathcal{D})) \le c \cdot d(\mathcal{D})$.

We define $n' \colon \mathbb{N} \to \mathbb{N}$ as $n'(d) = 2^{c\cdot d}$. We also define τ as $\tau(n,t) = \tau_{\mathrm{dom}}(\log n, \log n, t)$. For every $n \ge n'(d(\mathcal{D}))$, $i \le t_D(n)$, and $t \ge \tau(n, t_D(n))$, we have $\log n \ge c \cdot d(\mathcal{D}) \ge d(\mathcal{D}')$ and

$$t \ge \tau(n,t) = \tau_{\mathrm{dom}}(\log n, \log n, t) \ge \tau_{\mathrm{dom}}(d(\mathcal{D}'), \log n, t_D(n)).$$

Thus, for every $x \in \{0,1\}^*$ and every $i \le t_D(n)$, the following holds:

$$\begin{aligned}
\Pr_s[f_{\langle i,t\rangle}(s) = x_{[i]}] &= \Pr_s[U^t_{[i]}(s) = x_{[i]}] \\
&\ge \mathrm{Q}^t(x_{[i]}) \\
&\ge \frac{\mathcal{D}'_{n,i}(x_{[i]})}{2^{c\cdot d(\mathcal{D})} \cdot n^c} \\
&\ge \frac{\Pr_{x'\sim\mathcal{D}_n}[x'_{[i]} = x_{[i]}]}{n^{c+1}}.
\end{aligned}$$

Suppose that

$$\Pr_{x\sim\mathcal{D}_n}\left[\mathsf{L}_1\left(A'(x_{[i]}; 1^{\langle n,t,\epsilon^{-1},\delta^{-1}\rangle}), \mathsf{UnifInv}_{U^t_{[i]}}(x_{[i]})\right) > \epsilon\right] > \delta.$$

Then,

$$\begin{aligned}
&\Pr_s\left[\mathsf{L}_1\left(A(f_{\langle i,t\rangle}(s); 1^{\langle i,t\rangle}, 1^{\epsilon^{-1}}, 1^{n^{c+1}\delta^{-1}}), \mathsf{UnifInv}_{U^t_{[i]}}(f_{\langle i,t\rangle}(s))\right) > \epsilon\right] \\
&\quad\ge \frac{1}{n^{c+1}} \Pr_{x\sim\mathcal{D}_n}\left[\mathsf{L}_1\left(A(x_{[i]}; 1^{\langle i,t\rangle}, 1^{\epsilon^{-1}}, 1^{n^{c+1}\delta^{-1}}), \mathsf{UnifInv}_{U^t_{[i]}}(x_{[i]})\right) > \epsilon\right] \\
&\quad= \frac{1}{n^{c+1}} \Pr_{x\sim\mathcal{D}_n}\left[\mathsf{L}_1\left(A'(x_{[i]}; 1^{\langle n,t,\epsilon^{-1},\delta^{-1}\rangle}), \mathsf{UnifInv}_{U^t_{[i]}}(x_{[i]})\right) > \epsilon\right] \\
&\quad> \frac{\delta}{n^{c+1}}.
\end{aligned}$$

However, this contradicts the choice of the confidence parameter $n^{c+1}\delta^{-1}$ for A.

Hence, we conclude that

$$\Pr_{x\sim\mathcal{D}_n}\left[\mathsf{L}_1\left(A'(x_{[i]}; 1^{\langle n,t,\epsilon^{-1},\delta^{-1}\rangle}), \mathsf{UnifInv}_{U^t_{[i]}}(x_{[i]})\right) \le \epsilon\right] \ge 1 - \delta.$$

□

Now, we derive Theorem 4.7.1 from Lemma 4.7.3.

Proof of Theorem 4.7.1. Let n' and τ be the same functions as Lemma 4.7.3. Let A' be the randomized algorithm in Lemma 4.7.3. Then, the algorithm UE is constructed as

$$\mathsf{UE}(x; 1^{\langle n,k,t,\epsilon^{-1},\delta^{-1}\rangle}) = U^t(A'(x; 1^{\langle n,t,\epsilon^{-1},\delta^{-1}\rangle}))_{[i+1:i+k]},$$

where $i = |x|$. Furthermore, let $n_{\mathsf{UE}}(d) \equiv n'(d)$ and $\tau_{\mathsf{UE}}(n,t) \equiv \tau(n,t)$.

The correctness is verified as follows: for every $x \in \{0,1\}^*$ and every $i \le |x|$,

$$\begin{aligned}
&\mathsf{L}_1\left(\mathsf{UE}(x_{[i]}; 1^{\langle n,k,t,\epsilon^{-1},\delta^{-1}\rangle}), \mathsf{Next}_k(\mathrm{Q}^t, x_{[i]})\right) \\
&\quad = \mathsf{L}_1\left(U^t(A'(x_{[i]}; 1^{\langle n,t,\epsilon^{-1},\delta^{-1}\rangle}))_{[i+1:i+k]}, \mathsf{Next}_k(\mathrm{Q}^t, x_{[i]})\right) \\
&\quad = \mathsf{L}_1\left(U^t(A'(x_{[i]}; 1^{\langle n,t,\epsilon^{-1},\delta^{-1}\rangle}))_{[i+1:i+k]}, U^t(\mathsf{UnifInv}_{U^t_{[i]}}(x_{[i]}))_{[i+1:i+k]}\right) \\
&\quad \le \mathsf{L}_1\left(A'(x_{[i]}; 1^{\langle n,t,\epsilon^{-1},\delta^{-1}\rangle}), \mathsf{UnifInv}_{U^t_{[i]}}(x_{[i]})\right).
\end{aligned}$$

Therefore, for all $n, i, t, \epsilon^{-1}, \delta^{-1} \in \mathbb{N}$ with $n \ge n_0$ and $t \ge \tau_{\mathsf{UE}}(n, t_D(n)) = \tau(n, t_D(n))$,

$$\begin{aligned}
&\Pr_{x\sim\mathcal{D}_n}\left[\mathsf{L}_1\left(\mathsf{UE}(x_{[i]}; 1^{\langle n,k,t,\epsilon^{-1},\delta^{-1}\rangle}), \mathsf{Next}_k(\mathrm{Q}^t, x_{[i]})\right) \le \epsilon\right] \\
&\quad \ge \Pr_{x\sim\mathcal{D}_n}\left[\mathsf{L}_1\left(A'(x_{[i]}; 1^{\langle n,t,\epsilon^{-1},\delta^{-1}\rangle}), \mathsf{UnifInv}_{U^t_{[i]}}(x_{[i]})\right) \le \epsilon\right] \ge 1-\delta.
\end{aligned}$$

□

4.7.2 Proof by Estimating Universal a Priori Probability

We present another proof of Theorem 4.7.1, where we construct another extrapolation algorithm that predicts the next k bits one by one according to the approximated likelihood of the next 1 bit.

Proof of Theorem 4.7.1. Since there is no infinitely-often one-way function, there exists the algorithm M in Theorem 4.6.3. We construct the algorithm UE based on M as follows: On input $x, 1^{\langle n,k,t,\epsilon^{-1},\delta^{-1}\rangle}$, the algorithm UE samples $y_j \in \{0,1,\varepsilon\}$ inductively in $j \in [k]$ according to the following procedure: if $y_{j-1} = \varepsilon$ and $j \ge 2$, then $y_j = \varepsilon$; otherwise,

$$y_j = \begin{cases} 0 & \text{with probability } \min\{p_0/p_\varepsilon, 1\} \\ 1 & \text{with probability } \min\{p_1/p_\varepsilon, 1\} \\ \varepsilon & \text{with probability } \max\{(p_\varepsilon - p_0 - p_1)/p_\varepsilon, 0\} \end{cases}$$

where

$$\begin{aligned}
p_0 &= M(x \circ y_1 \circ \cdots \circ y_{j-1} \circ 0; 1^t, 1^{\delta'^{-1}}) \\
p_1 &= M(x \circ y_1 \circ \cdots \circ y_{j-1} \circ 1; 1^t, 1^{\delta'^{-1}}) \\
p_\varepsilon &= M(x \circ y_1 \circ \cdots \circ y_{j-1}; 1^t, 1^{\delta'^{-1}})
\end{aligned}$$

for $\delta' = \min\{\epsilon/(4k+\epsilon), \epsilon\delta/6kn^{c+1}\}$, where c is a sufficiently large constant specified later. Then, UE outputs $y_1 \circ \cdots \circ y_k$. If $p_\varepsilon = 0$ at some stage, then UE outputs "error" and halts.

We verify the correctness of UE according to the following three steps. First, we assume the ideal case in which M can output the exact value of $\mathrm{Q}^{t,*}(x)$ for any given x. Second, we take the multiplicative approximation error $(1 \pm \delta')$ into account. Finally, we take the confidence error δ' into account. Below, we omit the parameters to UE and M for readability.

Suppose that M can output the exact value of $\mathrm{Q}^{t,*}(x)$ for a given x with probability 1 over the choice of x and randomness for M. Then, by induction in j, we can easily verify that if $y_{j-1} \neq \varepsilon$, then for each $b \in \{0,1,\varepsilon\}$,

$$\Pr[y_j = b | y_1 \cdots y_{j-1}] = \Pr[\mathsf{Next}_1(\mathrm{Q}^t, xy_1 \cdots y_{j-1}) = b].$$

Thus, for every $y^* \in \{0,1\}^k$,

$$\begin{aligned}
\Pr[\mathsf{UE}(x) \text{ outputs } y^*] &= \prod_{j=1}^{k} \Pr[y_j = y^*_j | y_{[j-1]} = y^*_{[j-1]}] \\
&= \prod_{j=1}^{k} \Pr[\mathsf{Next}_1(\mathrm{Q}^t, xy^*_1 \cdots y^*_{j-1}) = y^*_j] \\
&= \Pr[\mathsf{Next}_k(\mathrm{Q}^t, x) = y^*],
\end{aligned}$$

and for every $y^* \in \{0,1\}^{k'}$ with $k' < k$,

$$\begin{aligned}
\Pr[\mathsf{UE}(x) \text{ outputs } y^*] &= \prod_{j=1}^{k'} \Pr[y_j = y^*_j | y_{[j-1]} = y^*_{[j-1]}] \cdot \Pr[y_{k'+1} = \varepsilon | y_{[k']} = y^*_{[k']}] \\
&= \prod_{j=1}^{k'} \Pr[\mathsf{Next}_1(\mathrm{Q}^t, xy^*_1 \cdots y^*_{j-1}) = y^*_j] \cdot \Pr[\mathsf{Next}_1(\mathrm{Q}^t, xy^*_1 \cdots y^*_{k'}) = \varepsilon] \\
&= \Pr[\mathsf{Next}_k(\mathrm{Q}^t, x) = y^*].
\end{aligned}$$

Therefore, the distribution of $\mathsf{UE}(x)$ is statistically equivalent to $\mathsf{Next}_k(\mathrm{Q}^t, x)$.

Next, we take the approximation error into account, i.e., we assume that $M(x)$ outputs a value of $p \in [\mathrm{Q}^{t,*}(x)(1 \pm \delta')]$ for any given x.

Fix $j \in [k]$ and $x \in \{0,1\}^*$ arbitrarily. Notice that $p_b \in [\mathrm{Q}^{t,*}(xy_1 \cdots y_{j-1}b)(1 \pm \delta')]$ for each $b \in \{0,1,\varepsilon\}$. Let $\mathcal{D}_j$ denote the distribution of y_j given $y_1 \cdots y_{j-1}$. Then, the following claim holds:

Claim 4.7.4.

$$\mathsf{L}_1(\mathcal{D}_j, \mathsf{Next}_1(\mathrm{Q}^t, xy_1 \cdots y_{j-1})) \leq \frac{\epsilon}{2k}$$

We defer the proof of Claim 4.7.4 to the end of the proof because it has no technical novelty.

By Claim 4.7.4, we have

$$\begin{aligned}
&\mathsf{L}_1(\mathcal{D}_1 \cdots \mathcal{D}_j, \mathsf{Next}_j(\mathrm{Q}^t, x)) \\
&\quad \leq \mathsf{L}_1(\mathcal{D}_1 \cdots \mathcal{D}_j, \mathcal{D}_1 \cdots \mathcal{D}_{j-1}\mathsf{Next}_1(\mathrm{Q}^t, x \circ \mathcal{D}_1 \cdots \mathcal{D}_{j-1})) \\
&\qquad\qquad + \mathsf{L}_1(\mathcal{D}_1 \cdots \mathcal{D}_{j-1}\mathsf{Next}_1(\mathrm{Q}^t, x \circ \mathcal{D}_1 \cdots \mathcal{D}_{j-1}), \mathsf{Next}_j(\mathrm{Q}^t, x)) \\
&\quad \leq \frac{\epsilon}{2k} + \mathsf{L}_1(\mathcal{D}_1 \cdots \mathcal{D}_{j-1}\mathsf{Next}_1(\mathrm{Q}^t, x \circ \mathcal{D}_1 \cdots \mathcal{D}_{j-1}), \mathsf{Next}_{j-1}(\mathrm{Q}^t, x)\mathsf{Next}_1(\mathrm{Q}^t, x \circ \mathsf{Next}_{j-1}(\mathrm{Q}^t, x))) \\
&\quad \leq \frac{\epsilon}{2k} + \mathsf{L}_1(\mathcal{D}_1 \cdots \mathcal{D}_{j-1}, \mathsf{Next}_{j-1}(\mathrm{Q}^t, x)).
\end{aligned}$$

By induction in $j \in [k]$, the total variation distance between $\mathsf{Next}_j(\mathrm{Q}^t, x)$ and the distribution $\mathcal{D}_1 \cdots \mathcal{D}_j$ of $y_1 \cdots y_j$ is at most $j \cdot \frac{\epsilon}{2k}$. By letting $j = k$, the total variation distance between $\mathsf{Next}_k(\mathrm{Q}^t, x)$ and $\mathsf{UE}(x)$ is at most $\epsilon/2$.

Finally, we take the confidence error into account. By the same argument as above, we can assume again that the accuracy error is $\epsilon = 0$ at first.

Let $\mathcal{D}' = \{D'_{n,i}\}_{n,i \in \mathbb{N}}$ be a distribution family, where each $\mathcal{D}'_{n,i}$ is a distribution of $x_{[i]}$ for $x \sim \mathcal{D}_n$. Then, $\mathcal{D}'$ is $O(t_D(n))$-time samplable, and $d(\mathcal{D}') = O(d(\mathcal{D}))$.

By Lemma 4.5.3, there exists a polynomial τ_{dom} such that for every $n, i \in \mathbb{N}$ with $i \leq t_D(n)$, every $t \geq \tau_{\mathrm{dom}}(d(\mathcal{D}'), \log n, t_D(n))$, and every $x \in \{0,1\}^*$, it holds that $\mathrm{Q}^t(x) \geq \mathcal{D}'_{n,i}(x)/(2^{c \cdot d(\mathcal{D})} \cdot n^c)$ by choosing sufficiently large $c \geq 1$. We can also assume that $d(\mathcal{D}') = O(d(\mathcal{D})) \leq c \cdot d(\mathcal{D})$.

We define $n' \colon \mathbb{N} \to \mathbb{N}$ as $n'(d) = 2^{c \cdot d}$. We also define τ as

$$\tau(n, t) = \tau_{\mathrm{dom}}(\log n, \log n, t).$$

For every $n \geq n'(d(\mathcal{D}))$, $i \leq t_D(n)$, and $t \geq \tau(n, t_D(n))$, we have $\log n \geq c \cdot d(\mathcal{D}) \geq d(\mathcal{D}')$ and

$$t \geq \tau(n, t_D(n)) = \tau_{\mathrm{dom}}(\log n, \log n, t_D(n)) \geq \tau_{\mathrm{dom}}(d(\mathcal{D}'), \log n, t_D(n)).$$

Thus, for every $x \in \{0,1\}^i$, the following holds:

$$\mathrm{Q}^t(x) \geq \frac{\mathcal{D}'_{n,i}(x)}{2^{c \cdot d(\mathcal{D})} \cdot n^c} \geq \frac{\mathcal{D}'_{n,i}(x)}{n^{c+1}}.$$

Furthermore, for every $x \in \{0,1\}^i$, every $j \in [k]$, and every $y_1, \ldots, y_{j-1} \in \{0,1\}$,

$$\begin{aligned}
\mathrm{Q}^t_{[i+j-1]}(x y_1 \cdots y_{j-1}) &= \mathrm{Q}^{t,*}(x) \cdot \Pr[\mathsf{Next}_{j-1}(\mathrm{Q}^t, x) = y_1 \cdots y_{j-1}] \\
&\geq \mathrm{Q}^t(x) \cdot \Pr[\mathsf{Next}_{j-1}(\mathrm{Q}^t, x) = y_1 \cdots y_{j-1}] \\
&\geq \frac{\mathcal{D}'_{n,i}(x)}{n^{c+1}} \cdot \Pr[\mathsf{Next}_{j-1}(\mathrm{Q}^t, x) = y_1 \cdots y_{j-1}].
\end{aligned}$$

Remember that at the j-th stage, the algorithm M is given $x y_1 \cdots y_j$, where $x \sim \mathcal{D}'_{n,i}$ and $y_1 \cdots y_{j-1} \sim \mathsf{Next}_{j-1}(\mathrm{Q}^t, x)$.

For every $b \in \{0, 1, \varepsilon\}$, the distribution of $x' \circ b$, where $x' \sim \mathrm{Q}^t_{[i+j-1]}$ is polynomial-time samplable. By Theorem 4.6.3, there exist $n_0 \in \mathbb{N}$ and a polynomial τ_0 such that for every $n \geq n_0$, $t \geq \tau_0(n)$, every $b \in \{0, 1, \varepsilon\}$,

$$\Pr_{x y_1 \cdots y_{j-1} \sim \mathrm{Q}^t_{[i+j-1]}} \left[M(x y_1 \cdots y_{j-1} b) \text{ fails to output } \mathrm{Q}^{t,*}(x y_1 \cdots y_{j-1} b) \right] < \delta' \leq \frac{\epsilon \delta}{6kn^{c+1}}.$$

Thus, for every $n \geq \max\{n_0, n'(d(\mathcal{D}))\}$ and $t \geq \max\{\tau_0(n), \tau(n, t_D(n))\}$,

$$\begin{aligned}
&\Pr_{x \sim \mathcal{D}'_{n,i}, y_1 \cdots y_{j-1} \sim \mathsf{Next}_{j-1}(\mathrm{Q}^t, x)} \left[M(x y_1 \cdots y_{j-1} b) \text{ fails to output } \mathrm{Q}^{t,*}(x y_1 \cdots y_{j-1} b) \right] \\
&\leq n^{c+1} \cdot \Pr_{x y_1 \cdots y_{j-1} \sim \mathrm{Q}^t} \left[M(x y_1 \cdots y_{j-1} b) \text{ fails to output } \mathrm{Q}^{t,*}(x y_1 \cdots y_{j-1} b) \right] \leq n^{c+1} \cdot \frac{\epsilon \delta}{6kn^{c+1}} = \frac{\epsilon \delta}{6k}.
\end{aligned}$$

Since UE executes M at most $3k$ times, by the union bound, it holds that

$$\Pr_{x \sim \mathcal{D}_n, M} \left[M \text{ fails at some stage in executing } \mathsf{UE}(x_{[i]}) \right] \leq \frac{\epsilon \delta}{2}.$$

By Markov's inequality,

$$\Pr_{x\sim\mathcal{D}_n}\left[\Pr_{M}\left[M \text{ fails at some stage in executing } \mathsf{UE}(x_{[i]})\right] \le \epsilon/2\right] \ge 1-\delta.$$

Now we consider the approximation error again. For every choice of x that satisfies the event above, (i) if M does not fail, then the total variation distance between $\mathsf{UE}(x_{[i]})$ and $\mathsf{Next}_k(\mathrm{Q}^t, x)$ is at most $\epsilon/2$; and (ii) the probability that M fails at some stage is at most $\epsilon/2$. Thus, we conclude that the total variation distance between $\mathsf{UE}(x_{[i]})$ and $\mathsf{Next}_k(\mathrm{Q}^t, x)$ is at most $\epsilon/2+\epsilon/2 = \epsilon$ for such x, i.e.,

$$\Pr_{x\sim\mathcal{D}_n}\left[\mathsf{L}_1(\mathsf{UE}(x_{[i]}), \mathsf{Next}_k(\mathrm{Q}^t, x_{[i]}) \le \epsilon\right] \ge 1-\delta.$$

Thus, the remaining is the proof of Claim 4.7.4.

Proof of Claim 4.7.4. For each $b \in \{0,1,\varepsilon\}$, let $p^*_b \in [0,1]$ be $p^*_b = \mathrm{Q}^{t,*}(xy_1\cdots y_{j-1}b)$. Remember that $p_b \in [p^*_b(1\pm\delta')]$. Therefore,

$$\begin{aligned}
\mathsf{L}_1(\mathcal{D}_i, \mathsf{Next}_1(\mathrm{Q}^t, xy_1\cdots y_{j-1})) &\le \frac{1}{2}\sum_{b\in\{0,1\}}\left|\frac{p_b}{p_\varepsilon} - \frac{p^*_b}{p^*_\varepsilon}\right| + \frac{1}{2}\left|\frac{p_\varepsilon - p_0 - p_1}{p_\varepsilon} - \frac{p^*_\varepsilon - p^*_0 - p^*_1}{p^*_\varepsilon}\right| \\
&= \frac{1}{2}\sum_{b\in\{0,1\}}\left|\frac{p_b}{p_\varepsilon} - \frac{p^*_b}{p^*_\varepsilon}\right| + \frac{1}{2}\left|\frac{p^*_0 + p^*_1}{p^*_\varepsilon} - \frac{p_0 + p_1}{p_\varepsilon}\right| \\
&\le \sum_{b\in\{0,1\}}\left|\frac{p_b}{p_\varepsilon} - \frac{p^*_b}{p^*_\varepsilon}\right|
\end{aligned}$$

For each $b \in \{0,1\}$, if $\frac{p_b}{p_\varepsilon} - \frac{p^*_b}{p^*_\varepsilon} \ge 0$, then

$$\left|\frac{p_b}{p_\varepsilon} - \frac{p^*_b}{p^*_\varepsilon}\right| = \frac{p_b}{p_\varepsilon} - \frac{p^*_b}{p^*_\varepsilon} \le \frac{(1+\delta')p^*_b}{(1-\delta')p^*_\varepsilon} - \frac{p^*_b}{p^*_\varepsilon} = \frac{p^*_b}{p^*_\varepsilon}\cdot\frac{2\delta'}{1-\delta'};$$

otherwise,

$$\left|\frac{p_b}{p_\varepsilon} - \frac{p^*_b}{p^*_\varepsilon}\right| = \frac{p^*_b}{p^*_\varepsilon} - \frac{p_b}{p_\varepsilon} \le \frac{p^*_b}{p^*_\varepsilon} - \frac{p^*_b(1-\delta')}{p^*_\varepsilon(1+\delta')} = \frac{p^*_b}{p^*_\varepsilon}\cdot\frac{2\delta'}{1+\delta'} \le \frac{p^*_b}{p^*_\varepsilon}\cdot\frac{2\delta'}{1-\delta'}.$$

In any case,

$$\left|\frac{p_b}{p_\varepsilon} - \frac{p^*_b}{p^*_\varepsilon}\right| \le \frac{p^*_b}{p^*_\varepsilon}\cdot\frac{2\delta'}{1-\delta'}.$$

Therefore,

$$\begin{aligned}
\mathsf{L}_1(\mathcal{D}_i, \mathsf{Next}_1(\mathrm{Q}^t, xy_1\cdots y_{j-1})) &\le \sum_{b\in\{0,1\}}\left|\frac{p_b}{p_\varepsilon} - \frac{p^*_b}{p^*_\varepsilon}\right| \\
&\le \sum_{b\in\{0,1\}}\frac{p^*_b}{p^*_\varepsilon}\cdot\frac{2\delta'}{1-\delta'} \\
&= \frac{2\delta'}{1-\delta'}\cdot\frac{p^*_0 + p^*_1}{p^*_\varepsilon} \\
&\le \frac{2\delta'}{1-\delta'} \\
&\le \frac{\epsilon}{2k},
\end{aligned}$$

where the last inequality is obtained by rearranging that $\delta' \leq \epsilon/(4k+\epsilon)$. □

□

4.8 Simulating Cheating Learners by Universal Extrapolation

In this section, we consider the online learning framework introduced in Section 4.3.3. Remember that, in the framework, a learner first observes advice information $x^i \in \{0,1\}^*$ and then obtains $y^i \in \{0,1\}^*$ (we call y^i the i-th label) at stage $i \in \mathbb{N}$, where each data may have correlation with the previous streams. The task of the learner at stage i is, for a given advice string x_i, to predict the next outcome y_i.

We introduce several notions to discuss the learning framework above more formally. In Section 4.8, we use $a \in \mathbb{N} \cup \{0\}$ and $b \in \mathbb{N}$ to represent the size of each observation (x^i, y^i) as $|x^i| = a$ and $|y^i| = b$, and we use $m \in \mathbb{N}$ to represent the total number of stages. Note that our results hold in more general cases in which $|x^i|$ and $|y^i|$ vary by the same proof.

For every offline stream $x \in \{0,1\}^*$, a stream $x^1, y^1, \ldots, x^m, y^m$ of data for online learning is determined as follows: for each $i \in \mathbb{N}$,

$$x^i = x_{[(a+b)(i-1)+1:(a+b)(i-1)+a]} \text{ and } y^i = y_{[(a+b)(i-1)+a+1:(a+b)i]}.$$

Notice that $x = x^1y^1 \circ \cdots \circ x^my^m$, and x^i and y^i can be empty when $|x| < m(a+b)$. For each $i \in [m]$, we let $x^{<i} := x^1y^1 \circ \cdots \circ x^{i-1}y^{i-1} = x_{[(a+b)(i-1)]}$. Furthermore, for every distribution $\mathcal{D}$ on (offline) binary strings, we let $\mathcal{D}^{<i}$ represent a distribution of $x^{<i}$ for $x \sim \mathcal{D}$ and let $\mathcal{D}^{i,x^{<i}}$ represent the conditional distribution of x'^i for $x' \sim \mathcal{D}$ given $x'^{<i} = x^{<i}$.

When the offline string x is selected according to some distribution $\mathcal{D}_{n,z}$ indexed by $n \in \mathbb{N}$ and $z \in \{0,1\}^*$, then the conditional distribution of the i-th label given $(x^{<i}, x^i)$ is $\mathsf{Next}_b(\mathcal{D}_{n,z}, x^{<i}x^i)$. When $\mathcal{D}_{n,z}$ and $x^{<i}$ are clear in context, we use the notation Label_i^{z,x^i} to refer to $\mathsf{Next}_b(\mathcal{D}_{n,z}, x^{<i}x^i)$.

Remember that a cheating learner is a learner that can freely observe the labels in the future by the additional oracle access to Label_i^{z,x^i}. The key insight for translating UE into learning algorithms in the framework above is that we can replace the oracle access to Label_i^{z,x^i} with UE in the average-case setting. This is stated as the following meta-theorem.

Theorem 4.8.1. *Suppose that* UE *in Theorem 4.7.1 exists. Then, for every oracle machine (i.e., a cheating learner)* $L^?_{cheat}$ *of polynomial-time computable query complexity* $q(\cdot)$*, there exist a randomized algorithm* L *and a polynomial* m_0 *satisfying the following: for every* $t_G(s)$*-time samplable family* $\mathcal{G} = \{\mathcal{G}_s\}_{s\in\mathbb{N}}$ *and every* $t_D(n,|z|)$*-time samplable family* $\mathcal{D} = \{\mathcal{D}_{n,z}\}_{n\in\mathbb{N},z\in\{0,1\}^*}$*, where each* $\mathcal{G}_s$ *is over* s*-bit strings, and each* $\mathcal{D}_{n,z}$ *is over binary strings, there exists* $n_0 \in \mathbb{N}$ *such that for every* $a \in \mathbb{N}\cup\{0\}$*, every* $n, s, b, t, \delta^{-1}, \lambda^{-1} \in \mathbb{N}$ *with* $n \geq n_0$ *and* $t \geq \max\{t_G(s), t_D(n,s)\}$*, every auxiliary input* $w \in \{0,1\}^*$*, and every* $m \geq m_0(\log n, s, q(w), \delta^{-1}, \lambda^{-1})$*,*

$$\Pr_{z,i,x^{<i},x^i}\left[\mathsf{L}_1\left(L(x^{<i}, x^i, w; 1^{\langle n,s,b,t,\delta^{-1},\lambda^{-1}\rangle}), L^{\mathsf{Label}_i^{z,x^i}}_{cheat}(w)\right) \leq \lambda\right] \geq 1-\delta,$$

where $z \sim \mathcal{G}_s, i \sim [m], x^{<i} \sim \mathcal{D}^{<i}_{n,z}$*, and* $x^i \sim \mathcal{D}^{i,x^{<i}}_{n,z}$*.*

Furthermore, $m_0(\log n, r, q, \delta^{-1}, \lambda^{-1}) = O((\log n + s)\cdot q \cdot \delta^{-1}\lambda^{-2})$*,* $n_0 = n'(d(\mathcal{G}), d(\mathcal{D}))$ *for a universal function* $n' \colon \mathbb{N}\times\mathbb{N} \to \mathbb{N}$*, and* L *halts in polynomial time in the input length and the running time of* L_{cheat}*.*

The proof of Theorem 4.8.1 is presented in Section 4.8.1. Note that Theorem 4.8.1 is sufficient for simulating any polynomial-time cheating learners with polynomial-time overhead. As applications, we show universal distributional learning and universal learning ACDs (i.e., Theorem 4.2.1) in Section 4.8.2, universal top-k prediction in Section 4.8.3, and universal likelihood estimation in Section 4.8.4.

4.8.1 A Proof of Meta-Theorem

First, we show the following key lemma, as outlined in Section 4.3.4.

Lemma 4.8.2. *For every distribution $\mathcal{D}$ over binary strings such that $\mathcal{D}$ has a t_D-time sampler described by d bits, and for every $a \in \mathbb{N} \cup \{0\}$, every $t, q, b, m \in \mathbb{N}$ with $t \geq \tau_{\mathrm{dom}}(d, t_D)$, and every q-query (possibility not efficiently computable) randomized oracle machine I,*

$$\mathop{\mathbb{E}}_{i\sim[m], x^{<i}\sim\mathcal{D}^{<i}, x^i\sim\mathcal{D}^{i,x^{<i}}} \left[\mathrm{KL}\left(I^{\mathsf{Next}_b(\mathcal{D},x^{<i}x^i)} || I^{\mathsf{Next}_b(\mathrm{Q}^t,x^{<i}x^i)}\right)\right] \leq q \cdot \frac{O(d)}{m},$$

where the hidden constant in $O(d)$ depends on only the universal Turing machine.

Proof. Fix $a \in \mathbb{N} \cup \{0\}$ and $b, m, q \in \mathbb{N}$ arbitrarily. For each $x \in \mathrm{supp}(\mathcal{D})$, we use the notations $x^1, y^1, \ldots, x^m, y^m$ and $x^{<i}$ as defined at the beginning of Section 4.8.

For every q-query randomized oracle machine I and every distribution $\mathcal{O}$, the distribution of $I^{\mathcal{O}}$ is considered as $I(a_1, \ldots, a_q)$ for $a_1, \ldots, a_q \sim \mathcal{O}$, where we regard I as a randomized function. Thus, for every distributions $\mathcal{O}$ and $\mathcal{O}'$ with $\mathrm{KL}(\mathcal{O}; \mathcal{O}') < \infty$, we have

$$\mathrm{KL}(I^{\mathcal{O}} || I^{\mathcal{O}'}) \leq \mathrm{KL}(I(\mathcal{O}_1, \ldots, \mathcal{O}_q) || I(\mathcal{O}'_1, \ldots, \mathcal{O}'_q)) \leq \mathrm{KL}(\mathcal{O}_1, \ldots, \mathcal{O}_q || \mathcal{O}'_1, \ldots, \mathcal{O}'_q) = q \cdot \mathrm{KL}(\mathcal{O} || \mathcal{O}),$$

where each $\mathcal{O}_i$ (resp. $\mathcal{O}'_i$) is an independent random variable drawn from $\mathcal{O}$ (resp. $\mathcal{O}'$), and the second inequality follows from Fact 4.5.2.

If $t \geq \tau_{\mathrm{dom}}(d, t_D)$, then $\mathrm{Q}^t(x) \geq \mathcal{D}(x)/2^{O(d)}$ for every $x \in \{0,1\}^*$ by Lemma 4.5.3; thus, for every $x \in \{0,1\}^*$, $\mathrm{KL}(\mathsf{Next}_b(\mathcal{D}, x) || \mathsf{Next}_b(\mathrm{Q}^t, x)) < \infty$.

Therefore,

$$\begin{aligned}
&\mathop{\mathbb{E}}_{i\sim[m], x^{<i}\sim\mathcal{D}^{<i}, x^i\sim\mathcal{D}^{i,x^{<i}}} \left[\mathrm{KL}\left(I^{\mathsf{Next}_b(\mathcal{D},x^{<i}x^i)} || I^{\mathsf{Next}_b(\mathrm{Q}^t,x^{<i}x^i)}\right)\right] \\
&= \frac{q}{m} \sum_{i=1}^{m} \sum_{x^{<i}\in\mathrm{supp}(\mathcal{D}^{<i})} \sum_{x^i\in\mathrm{supp}(\mathcal{D}^{<i,x^i})} \mathcal{D}^*(x^{<i}x^i) \cdot \mathrm{KL}(\mathsf{Next}_b(\mathcal{D}, x^{<i}x^i) || \mathsf{Next}_b(\mathrm{Q}^t, x^{<i}x^i)). \qquad (4.3)
\end{aligned}$$

For readability, we let $\mathcal{D}(y|x^{<i}x^i)$ (resp. $\mathrm{Q}^t(y|x^{<i}x^i)$) denote $\mathsf{Next}_b(\mathcal{D}, x^{<i}x^i)(y)$ (resp. $\mathsf{Next}_b(\mathrm{Q}^t, x^{<i}x^i)(y)$)

for each $i \in [m]$ and $y \in \{0,1\}^{\leq b}$. Then, we have that, for every $i \in [m]$,

$$\begin{aligned}
&\sum_{x^{<i},x^i} \mathcal{D}^*(x^{<i}x^i) \cdot \mathrm{KL}(\mathsf{Next}_b(\mathcal{D}, x^{<i}x^i)||\mathsf{Next}_b(\mathrm{Q}^t, x^{<i}x^i)) \\
&\quad = \sum_{x^{<i},x^i} \sum_{y^i \in \mathrm{supp}(\mathsf{Next}_b(\mathcal{D},x^{<i}x^i))} \mathcal{D}^*(x^{<i}x^i) \cdot \mathcal{D}(y^i|x^{<i}x^i) \log \frac{\mathcal{D}(y^i|x^{<i}x^i)}{\mathrm{Q}^t(y^i|x^{<i}x^i)} \\
&\quad = \sum_{x^{<i},x^i} \sum_{y^i \in \mathrm{supp}(\mathsf{Next}_b(\mathcal{D},x^{<i}x^i))} \mathcal{D}^*(x^{<i}x^iy^i) \log \frac{\mathcal{D}(y^i|x^{<i}x^i)}{\mathrm{Q}^t(y^i|x^{<i}x^i)} \\
&\quad = \sum_{x^{<i+1} \in \mathrm{supp}(\mathcal{D}^{<i+1})} \mathcal{D}^*(x^{<i+1}) \log \frac{\mathcal{D}(y^i|x^{<i}x^i)}{\mathrm{Q}^t(y^i|x^{<i}x^i)} \\
&\quad = \sum_{x \in \mathrm{supp}(\mathcal{D})} \mathcal{D}(x) \log \frac{\mathcal{D}(y^i|x^{<i}x^i)}{\mathrm{Q}^t(y^i|x^{<i}x^i)}.
\end{aligned}$$

Therefore, by inequality (4.3),

$$\begin{aligned}
\mathop{\mathbb{E}}_{i,x^{<i},x^i}\left[\mathrm{KL}\left(I^{\mathsf{Next}_b(\mathcal{D},x^{<i}x^i)}||I^{\mathsf{Next}_b(\mathrm{Q}^t,x^{<i}x^i)}\right)\right] &= \frac{q}{m}\sum_{i=1}^{m}\sum_{x\in\mathrm{supp}(\mathcal{D})} \mathcal{D}(x)\log\frac{\mathcal{D}(y^i|x^{<i}x^i)}{\mathrm{Q}^t(y^i|x^{<i}x^i)} \\
&= \frac{q}{m}\sum_{x\in\mathrm{supp}(\mathcal{D})} \mathcal{D}(x)\sum_{i=1}^{m}\log\frac{\mathcal{D}(y^i|x^{<i}x^i)}{\mathrm{Q}^t(y^i|x^{<i}x^i)} \\
&= \frac{q}{m}\sum_{x\in\mathrm{supp}(\mathcal{D})} \mathcal{D}(x)\log\frac{\prod_{i=1}^{m}\mathcal{D}(y^i|x^{<i}x^i)}{\prod_{i=1}^{m}\mathrm{Q}^t(y^i|x^{<i}x^i)} \\
&= \frac{q}{m}\sum_{x\in\mathrm{supp}(\mathcal{D})} \mathcal{D}(x)\log\frac{\mathcal{D}(x)}{\mathrm{Q}^t(x)}\cdot\frac{\mathrm{Q}^{t,*}(x^1,\text{-},\ldots,x^m,\text{-})}{\mathcal{D}^*(x^1,\text{-},\ldots,x^m,\text{-})},
\end{aligned}$$

where $\mathrm{Q}^{t,*}(x^1,\text{-},\ldots,x^m,\text{-})$ (resp. $\mathcal{D}^*(x^1,\text{-},\ldots,x^m,\text{-})$) represents the probability that a sampling according to Q^t (resp. $\mathcal{D}$) is consistent with $x^1,\ldots,x^m$.

By the assumption that $t \geq \tau_{\mathrm{dom}}(d, t_D)$, Lemma 4.5.3 implies that $\mathrm{Q}^t(x) \geq \mathcal{D}(x)/2^{O(d)}$, i.e.,

$$\log\frac{\mathcal{D}(x)}{\mathrm{Q}^t(x)} \leq O(d),$$

where we implicitly use the fact that $\mathrm{Q}^t(x) \geq \mathcal{D}(x)/2^{O(d)} > 0$ for each $x \in \mathrm{supp}(\mathcal{D})$.

Let S be a set defined as $S = \{(x^1,\ldots,x^m) : x \in \mathrm{supp}(\mathcal{D})\}$. Then, the lemma is derived as follows:

$$\begin{aligned}
\mathop{\mathbb{E}}_{i,x^{<i},x^i}\left[\mathrm{KL}\left(I^{\mathsf{Next}_b(\mathcal{D},x^{<i}x^i)}||I^{\mathsf{Next}_b(\mathrm{Q}^t,x^{<i}x^i)}\right)\right] &= \frac{q}{m}\sum_{x\in\mathrm{supp}(\mathcal{D})} \mathcal{D}(x)\log\frac{\mathcal{D}(x)}{\mathrm{Q}^t(x)}\cdot\frac{\mathrm{Q}^{t,*}(x^1,\text{-},\ldots,x^m,\text{-})}{\mathcal{D}^*(x^1,\text{-},\ldots,x^m,\text{-})} \\
&\leq \frac{q}{m}\sum_{x\in\mathrm{supp}(\mathcal{D})} \mathcal{D}(x)\left(O(d)+\log\frac{\mathrm{Q}^{t,*}(x^1,\text{-},\ldots,x^m,\text{-})}{\mathcal{D}^*(x^1,\text{-},\ldots,x^m,\text{-})}\right) \\
&= q\cdot\frac{O(d)}{m} + \sum_{x\in\mathrm{supp}(\mathcal{D})} \mathcal{D}(x)\log\frac{\mathrm{Q}^{t,*}(x^1,\text{-},\ldots,x^m,\text{-})}{\mathcal{D}^*(x^1,\text{-},\ldots,x^m,\text{-})}
\end{aligned}$$

$$
\begin{aligned}
&= q \cdot \frac{O(d)}{m} + \sum_{(x^1,\dots,x^m)\in S} \mathcal{D}^*(x^1,\text{-},\dots,x^m,\text{-}) \log \frac{\mathrm{Q}^{t,*}(x^1,\text{-},\dots,x^m,\text{-})}{\mathcal{D}^*(x^1,\text{-},\dots,x^m,\text{-})} \\
&\le q \cdot \frac{O(d)}{m} + \log \sum_{(x^1,\dots,x^m)\in S} \mathcal{D}^*(x^1,\text{-},\dots,x^m,\text{-}) \frac{\mathrm{Q}^{t,*}(x^1,\text{-},\dots,x^m,\text{-})}{\mathcal{D}^*(x^1,\text{-},\dots,x^m,\text{-})} \\
&= q \cdot \frac{O(d)}{m} + \log \sum_{(x^1,\dots,x^m)\in S} \mathrm{Q}^{t,*}(x^1,\text{-},\dots,x^m,\text{-}) \\
&\le q \cdot \frac{O(d)}{m} + \log 1 \\
&= q \cdot \frac{O(d)}{m},
\end{aligned}
$$

where the second inequality follows from the Jensen's inequality. □

Theorem 4.8.1 follows from Theorem 4.7.1 and Lemma 4.8.2.

Proof of Theorem 4.8.1. Let UE be the universal extrapolation algorithm in Theorem 4.7.1. Let n_{UE} and τ_{UE} be the functions in Theorem 4.7.1. Let $L^{?}_{cheat}$ be an arbitrary oracle machine (a cheating learner) of polynomial-time computable query complexity $q := q(w)$.

For every $t_G(s)$-time samplable distribution family $\mathcal{G}$ and every $t_D(n,|z|)$-time samplable distribution family $\mathcal{D}$, we define another distribution family $\mathcal{D}' = \{\mathcal{D}'_{n'}\}_{n'\in\mathbb{N}}$ as $\mathcal{D}'_{n'} \equiv \mathcal{D}_{n,\mathcal{G}_s}$ for every $n' = \langle n,s\rangle$. It holds that $d(\mathcal{D}') = O(d(\mathcal{G}) + d(\mathcal{D}))$. Let $n_1 \colon \mathbb{N}\times\mathbb{N}\to\mathbb{N}$ be a function such that $n_1(d(\mathcal{G}),d(\mathcal{D})) = n_{\mathsf{UE}}(d(\mathcal{D}')) = 2^{O(d(\mathcal{G})+d(\mathcal{D}))}$. Furthermore, we select a sufficiently large polynomial $\tau_0(t,t')$ so that for every $n,s\in\mathbb{N}$, the distribution $\mathcal{D}_{\langle n,s\rangle}$ is samplable in time $\tau_0(t_G(s),t_D(n,s))$.

We construct the learner L that executes L_{cheat}, where the query access is simulated by UE. The formal construction is as follows: On input $x^{<i}, x^i, w, 1^{\langle n,s,b,t,\delta^{-1},\lambda^{-1}\rangle}$, the learner L executes the cheating learner $L^{?}_{cheat}(w)$, where L answers each query access to Label^{z,x^i}_i by

$$\mathsf{ans} \leftarrow \mathsf{UE}(x^{<i}x^i; 1^{\langle\langle n,s\rangle,b,t',\lambda'^{-1},2\delta^{-1}\rangle})$$

for $t' = \max\{\tau_{\mathsf{UE}}(\langle n,s\rangle,\tau_0(t,t)),\tau_1(n,s,t)\}$ and $\lambda' = \lambda/(2q(w))$ (with fresh randomness for each access), where τ_1 is a polynomial specified later. It is easy to verify that L halts in polynomial time in the input length and the running time of $L_{cheat}(w)$. Below, we show the correctness of L. For readability, we omit parameters for L and UE.

Let D be the sampler for $\mathcal{D}$ described by $d(\mathcal{D})$ bits. We choose a sufficiently large constant $c\ge 1$ so that (i) for every $d,d'\in\mathbb{N}$, $n_1(d,d')\le 2^{c(d+d')}$, and (ii) for every $n,s\in\mathbb{N}$ and $z\in\{0,1\}^s$, the sampler $D(1^n,z;\text{-})$ is described by $c(d(\mathcal{D})+\log n+s)$ bits. Let $n'(d,d') := 2^{c(d+d')} \ge n_1(d,d')$ and $n_0 := n'(d(\mathcal{G}),d(\mathcal{D}))$.

By Lemma 4.8.2, there exists a polynomial τ_1 such that for every $a\in\mathbb{N}\cup\{0\}$, every $n,s,t,m,b\in\mathbb{N}$, every $z\in\{0,1\}^s$ with $n\ge n_0$, $t\ge\tau_1(n,s,t_D(n,s))$, and every $w\in\{0,1\}^*$,

$$
\begin{aligned}
\mathop{\mathbb{E}}_{i\sim[m],x^{<i}\sim\mathcal{D}^{<i}_{n,z},x^i\sim\mathcal{D}^{i,x^{<i}}_{n,z}} \left[\mathrm{KL}(L^{\mathsf{Label}^{z,x^i}_i}_{cheat}(w)||L^{\mathsf{Next}_b(\mathrm{Q}^t,x^{<i}x^i)}_{cheat}(w))\right] &\le \frac{q\cdot O(d(\mathcal{D})+\log n+s)}{m} \\
&\le \frac{q\cdot c'(s+\log n)}{m},
\end{aligned}
$$

for some universal constant $c'>0$, where we use the fact that $\log n \ge \log n_0 \ge d(\mathcal{D})$.

Let $m_0 := \frac{q \cdot 4c'(s+\log n)}{\lambda^2 \delta} = O(\frac{q \cdot (s+\log n)}{\lambda^2 \delta})$. Then, for every $m \geq m_0$, $z \in \{0,1\}^s$, and $w \in \{0,1\}^*$, we have

$$\mathop{\mathbb{E}}_{i, x^{<i}, x^i} \left[\mathrm{KL}(L_{cheat}^{\mathsf{Label}_i^{z,x^i}}(w) || L_{cheat}^{\mathsf{Next}_b(\mathrm{Q}^t, x^{<i}x^i)}(w)) \right] \leq \frac{q \cdot c'(s+\log n)}{m} \leq \frac{\lambda^2 \delta}{4}.$$

By the non-negativity of KL divergence and Markov's inequality,

$$\Pr_{i, x^{<i}, x^i} \left[\mathrm{KL}(L_{cheat}^{\mathsf{Label}_i^{z,x^i}}(w) || L_{cheat}^{\mathsf{Next}_b(\mathrm{Q}^t, x^{<i}x^i)}(w)) > \frac{\lambda^2}{2} \right] < \frac{\delta}{2}.$$

By Pinsker's inequality (Fact 4.5.1),

$$\begin{aligned} &\Pr_{i, x^{<i}, x^i} \left[\mathsf{L}_1(L_{cheat}^{\mathsf{Label}_i^{z,x^i}}(w), L_{cheat}^{\mathsf{Next}_b(\mathrm{Q}^t, x^{<i}x^i)}(w)) \leq \frac{\lambda}{2} \right] \\ &\qquad \geq \Pr_{i, x^{<i}, x^i} \left[\mathrm{KL}(L_{cheat}^{\mathsf{Label}_i^{z,x^i}}(w) || L_{cheat}^{\mathsf{Next}_b(\mathrm{Q}^t, x^{<i}x^i)}(w)) \leq \frac{\lambda^2}{2} \right] \geq 1 - \frac{\delta}{2}. \end{aligned} \tag{4.4}$$

We remark that the above holds for any $a \in \mathbb{N} \cup \{0\}$, any $n, s, b, t, m, \delta^{-1}, \lambda^{-1} \in \mathbb{N}$, any $z \in \{0,1\}^s$, and any $w \in \{0,1\}^*$ satisfying that $n \geq n_0$, $t \geq \tau_1(n, s, t_D(n,s))$, and $m \geq m_0$.

For all $a \in \mathbb{N} \cup \{0\}$, $n, s, m, t, b, \delta^{-1}, \lambda^{-1} \in \mathbb{N}$, $i \in [m]$, and $w \in \{0,1\}^*$ with $n \geq n_0$, $m \geq m_0$, and $t \geq \max\{t_G(s), t_D(n,s)\}$, we have that $t' \geq \tau_{\mathsf{UE}}(\langle n, s\rangle, \tau_0(t,t)) \geq \tau_{\mathsf{UE}}(\langle n, s\rangle, \tau_0(t_G(s), t_D(n,s)))$, and by Theorem 4.7.1,

$$\Pr_{z \sim \mathcal{G}_s, x^{<i} \sim \mathcal{D}_{n,z}^{<i}, x^i \sim \mathcal{D}_{n,z}^{i, x^{<i}}} \left[\mathsf{L}_1 \left(\mathsf{UE}(x^{<i}x^i; 1^{\langle \langle n,s\rangle, b, t', \lambda'^{-1}, 2\delta^{-1}\rangle}), \mathsf{Next}_b(\mathrm{Q}^{t'}, x^{<i}x^i) \right) \leq \frac{\lambda}{2q} \right] \geq 1 - \frac{\delta}{2}. \tag{4.5}$$

Remember that (i) L simulates the oracle access by $\mathsf{UE}(x^{<i}x^i)$; and (ii) L_{cheat} makes the oracle access at most q times. Thus, for every $i, x^{<i}, x^i$ satisfying the event in inequality (4.5),

$$\mathsf{L}_1(L(x^{<i}, x^i, w), L_{cheat}^{\mathsf{Next}_b(\mathrm{Q}^{t'}, x^{<i}x^i)}(w)) = \mathsf{L}_1(L_{cheat}^{\mathsf{UE}(x^{<i}x^i)}(w), L_{cheat}^{\mathsf{Next}_b(\mathrm{Q}^{t'}, x^{<i}x^i)}(w)) \leq q \cdot \frac{\lambda}{2q} = \frac{\lambda}{2}.$$

Since $t' \geq \tau_1(n, s, t) \geq \tau_1(n, s, t_D(n,s))$, by inequalities (4.4) and (4.5) and the union bound, (i) $\mathsf{L}_1(L_{cheat}^{\mathsf{Label}_i^{z,x^i}}(w), L_{cheat}^{\mathsf{Next}_b(\mathrm{Q}^{t'}, x^{<i}x^i)}(w)) \leq \frac{\lambda}{2}$ and (ii) $\mathsf{L}_1(L(x^{<i}, x^i, w), L_{cheat}^{\mathsf{Next}_b(\mathrm{Q}^{t'}, x^{<i}x^i)}(w)) \leq \frac{\lambda}{2}$ hold with probability at least $1 - \delta$ over the choice of $z \sim \mathcal{G}_s, i \sim [m], x^{<i} \sim \mathcal{D}_{n,z}^{<i}$ and $x^i \sim \mathcal{D}_{n,z}^{i,x^{<i}}$. In this case, we have

$$\begin{aligned} &\mathsf{L}_1 \left(L(x^{<i}, x^i, w), L_{cheat}^{\mathsf{Label}_i^{z,x^i}}(w) \right) \\ &\quad \leq \mathsf{L}_1 \left(L(x^{<i}, x^i, w), L_{cheat}^{\mathsf{Next}_b(\mathrm{Q}^{t'}, x^{<i}x^i)}(w) \right) + \mathsf{L}_1 \left(L_{cheat}^{\mathsf{Next}_b(\mathrm{Q}^{t'}, x^{<i}x^i)}(w), L_{cheat}^{\mathsf{Label}_i^{z,x^i}}(w) \right) \\ &\quad \leq \frac{\lambda}{2} + \frac{\lambda}{2} = \lambda. \end{aligned}$$

Thus, we conclude that

$$\Pr_{z, i, x^{<i}, x^i} \left[\mathsf{L}_1 \left(L(x^{<i}, x^i, w), L_{cheat}^{\mathsf{Label}_i^{z,x^i}}(w) \right) \leq \lambda \right] \geq 1 - \delta.$$

□

4.8.2 Universal Distributional Learning and Universal Learning ACDs

In this section, we consider the problems of learning unknown distributions from samples, which was first studied by Kearns, Mansour, Ron, Rubinfeld, Schapire, and Sellie [KMRRSS94] (see also [Xia10]).

First, we formally introduce the learning models, i.e., a natural average-case variant of distributional learning and the problem of learning ACDs introduced by Naor and Rothblum [NR06].

Distributional Learning. We define a sampler of sample size n as a multi-output algorithm that is given a random seed as input and outputs an n-bit string. For convenience, we identify a sampler $S\colon \{0,1\}^\ell \to \{0,1\}^n$ with a distribution of $S(r)$ for $r \sim \{0,1\}^\ell$. For each sampler S, we define an example oracle EX_S as the oracle that returns $x \sim S$ for each access. For simplicity, we define the time complexity of sampler as a function in the sample size n instead of the seed length ℓ. For any $t, s \in \mathbb{N}$, we say that a sampler S of sample size n is t/s-time computable if there exists a program $\Pi_S \in \{0,1\}^{\leq s}$ such that $U^t(\Pi_S, r) = S(r_{[\ell]})$ for each seed $r \sim \{0,1\}^t$.

Informally, a distributional learner for t/s-time samplable distributions is given oracle access to EX_S for an unknown t/s-time computable sampler S and tries to construct a sampler that statistically simulates S. To consider the average-case setting, we define *a distribution on samplers* as a family $\mathcal{G} = \{\mathcal{G}_n\}_{n\in\mathbb{N}}$ of distributions, where each $\mathcal{G}_n$ is a distribution on descriptions of samplers of sample size n. For every distribution $\mathcal{G}$ on $t(n)/s(n)$-time computable samplers and every $n \in \mathbb{N}$, we use the notation $\mathcal{G}_n$ to refer to the n-th distribution in $\mathcal{G}$, i.e., the distribution on (at most $s(n)$-bit) descriptions of a $t(n)/s(n)$-time sampler of sample size n.

We define the average-case variant of distributional learning [KMRRSS94] as follows.

Definition 4.8.3 (Distributional learning on average)**.** *Let $\mathcal{C}$ be a class of distributions on the class $\mathcal{S}$ of samplers. We say that $\mathcal{S}$ is distributionally learnable in polynomial time on average under $\mathcal{C}$ if there exists a polynomial-time randomized oracle machine (i.e., learner) L such that for every distribution $\mathcal{G} \in \mathcal{C}$ (note that $\mathcal{G}$ is a distribution on samplers), for every sufficiently large $n \in \mathbb{N}$, and every $\epsilon^{-1}, \delta^{-1} \in \mathbb{N}$, the algorithm L satisfies that*

$$\Pr_{S_n\sim\mathcal{G}_n,\mathsf{EX}_{S_n},L}\left[L^{\mathsf{EX}_{S_n}}(1^n, 1^{\epsilon^{-1}}, 1^{\delta^{-1}}) \text{ outputs a circuit } h \text{ s.t. } \mathsf{L}_1(S_n, h(r)) \leq \epsilon\right] \geq 1-\delta,$$

where r is a uniformly random seed for h. We also define the sample complexity $m(n,\epsilon,\delta)$ as the upper bound of the number of oracle access $L(1^n, 1^{\epsilon^{-1}}, 1^{\delta^{-1}})$ requires.

Note that the learner L knows neither a target sampler S_n nor underlying distribution $\mathcal{D}$ except the prior knowledge of the classes $\mathcal{C}$ and $\mathcal{S}$ (i.e., a modeling of environment).

Learning Adaptively Changing Distributions. Next, we introduce learning ACDs first studied in [NR06].

An ACD (adaptively changing distribution) is a pair (G, D) of randomized Turing machines satisfying the following syntax: For every sample size $n \in \mathbb{N}$,

1. $G(1^n)$ randomly selects an initial state $s_0 \in \{0,1\}^*$.
2. For any $\sigma \in \{0,1\}^*$ that represents a current state, the algorithm $D(1^n, \sigma)$ randomly generates a sample $x \in \{0,1\}^n$ and a next state $s' \in \{0,1\}^*$ (Note that x and s' can be correlated).

Then, any ACD (G, D) determines an example oracle $\mathsf{EX}_{n,s_0,D}$ for each sample size $n \in \mathbb{N}$ and for each initial state s_0 generated by $G(1^n)$ as follows:

1. $\mathsf{EX}_{n,s_0,D}$ has a hidden internal state σ, which is initialized by s_0.

2. For each query access (without input), $\mathsf{EX}_{n,s_0,D}$ generates $(x, s') \leftarrow A(1^n, \sigma)$ and returns x as a sample. Then, $\mathsf{EX}_{n,s_0,D}$ updates the internal state σ as $\sigma := s'$.

For every functions $s(n)$ and $t(n)$, we say that an ACD (G, D) is $t(n)$-time samplable and has an $s(n)$-bit initial state if for every $n \in \mathbb{N}$, (i) $G(1^n)$ selects an initial state from $\{0,1\}^{\leq s(n)}$ in $t(n)$ time; and (ii) for every possible state σ, $D(1^n, \sigma)$ halts in $t(n)$ time (i.e., $\sigma \in \{0,1\}^{\leq t(n)}$).

Informally, in learning ACD (G, D), a learner has query access to $\mathsf{EX}_{n,s_0,D}$ for a given parameter 1^n, where s_0 is a hidden initial state selected by $G(1^n)$. The goal of learner is to choose some stage $i \in \mathbb{N}$ and, after observing the first i samples $x^1, \ldots, x^i$ from $\mathsf{EX}_{n,s_0,D}$, to statistically simulates the conditional distribution of the next sample x^{i+1} given the initial state s_0 and $x^1, \ldots, x^i$. For convenience, we use the notation $D_i^{s_0}(x^1, \ldots, x^i)$ to refer to the conditional distribution the learner tries to simulate at stage i.

Now, we present the formal definition of the learning model. Naor and Rothblum [NR06] considered the case in which a learner knows (G, D); while, we consider the more general case in which a learner does not know (G, D). When (G, D) is known, then the task of learning ACDs can be regarded as learning initial state s_0. However, when (G, D) is unknown, the task seems to become much more complicated, as seen in Sections 4.2 and 4.3.5. Particularly, by the argument in Section 4.3.5, even if the learner *knows* the initial state s_0, it does not mean that the learner can immediately simulates $D_i^{s_0}(x^1, \ldots, x^i)$.

Definition 4.8.4 (Universal learning ACDs)**.** *Let $s(n)$ and $t(n)$ be polynomials. We say that $t(n)$-time samplable ACDs of $s(n)$-bit initial state are universally learnable in polynomial time if there exists a randomized polynomial-time algorithm L such that for every $t(n)$-time samplable ACD (G, D) of $s(n)$-bit initial state, for every sufficiently large $n \in \mathbb{N}$, and every $\epsilon^{-1}, \delta^{-1} \in \mathbb{N}$, the algorithm L satisfies the following with probability at least $1 - \delta$ over the choice of $s_0 \sim G(1^n)$, samples from $\mathsf{EX}_{n,s_0,D}$, and randomness for L:*

1. *$L^{\mathsf{EX}_{n,s_0,D}}(1^n, 1^{\epsilon^{-1}}, 1^{\delta^{-1}})$ obtains samples $x^1, x^2, \ldots,$ from $\mathsf{EX}_{n,s_0,D}$.*

2. *After obtaining i samples $x^1, \ldots, x^i$ (where i is chosen by L), $L^{\mathsf{EX}_{n,s_0,D}}(1^n, 1^{\epsilon^{-1}}, 1^{\delta^{-1}})$ outputs some circuit h as a hypothesis without additional access to $\mathsf{EX}_{n,s_0,D}$.*

3. *The hypothesis h satisfies that $\mathsf{L}_1(D_i^{s_0}(x^1, \ldots, x^i), h(r)) \leq \epsilon$, where r is a uniformly random seed for h.*

We define the sample complexity $m(n, \epsilon, \delta)$ as the upper bound of the number of oracle access by $L(1^n, 1^{\epsilon^{-1}}, 1^{\delta^{-1}})$.

Note that average-case distributional learning in Definition 4.8.3 is a special case of universal learning ACDs in Definition 4.8.4, where an initial state s_0 is a target sampler, a generator G of ACD is a sampling algorithm for selecting the target sampler, and a sampling algorithm D of ACD does not change the internal state (i.e., always outputs $\sigma = s_0$ for a given current state $\sigma = s_0$).

Now, we show the following learnability result as an application of Theorem 4.8.1.

Theorem 4.8.5. *The following are equivalent:*

1. *There is no infinitely-often one-way function.*

2. (Universal Average-Case Distributional Learning) *For every polynomials $s(n), t(n)$ and $t'(n)$, $t(n)/s(n)$-time samplable distributions (i.e., $t(n)/s(n)$-time computable samplers) are distributionally learnable in polynomial time on average under (unknown) $t'(n)$-time samplable distributions with sample complexity $O((s(n)+\log n)\cdot\epsilon^{-2}\delta^{-1})$.*

3. (Universal Learning ACDs) *For every polynomials $s(n)$ and $t(n)$, $t(n)$-time samplable ACDs of $s(n)$-bit initial state are universally learnable in polynomial time with sample complexity $O((s(n)+\log n)\cdot\epsilon^{-2}\delta^{-1})$.*

We remark that the dependences of the confidence error δ^{-1} in sample complexity in item 3 of Theorem 4.8.5 is improved from δ^{-2} in [NR06].

Proof. (item 3 $\Rightarrow$ item 2) holds since distributional learning is a special case of learning ACDs. In fact, the following proof of (item 1 $\Rightarrow$ item 3) works even for (item 1 $\Rightarrow$ item 2).

(item 2 $\Rightarrow$ item 1) is due to the observation in [KMRRSS94, Theorem 17]. Particularly, an efficient distributional learner can distinguish any infinitely-often pseudorandom function $f=\{f_n:\{0,1\}^s\times\{0,1\}^n\to\{0,1\}^n\}$ from truly random function ϕ_n by distributionally learning the distribution of $x\circ f_n(r,x)$ for $x\sim\{0,1\}^n$. This is because the distribution of $x\circ\phi_n(x)$ cannot be statistically approximated by polynomial-size circuits with high probability. Note that this matches our average-case framework because the secret seed r for f is selected uniformly at random.

We derive (item 1 $\Rightarrow$ item 3) from Theorems 4.7.1 and 4.8.1. By the non-existence of infinitely-often one-way functions and Theorem 4.7.1, there exists the universal extrapolation algorithm UE. Therefore, we can apply Theorem 4.8.1.

We consider the trivial 1-query cheating learner $L^?_{cheat}$ that directly outputs a sample x obtained from the oracle. We apply Theorem 4.8.1 for $L^?_{cheat}$ and obtains a learner L that simulates $L^?_{cheat}$ as in Theorem 4.8.1. Note that L halts in polynomial time in the length of its input.

We construct a learner L' that learns $t(n)$-time samplable ACDs of $s(n)$-bit initial state as follows: On input $1^n, 1^{\epsilon^{-1}}$, and $1^{\delta^{-1}}$, the learner L' selects $i\sim[m_0]$, where $m_0:=m_0(n,\epsilon^{-1},\delta^{-1})=O((s(n)+\log n)\delta\epsilon^{-2})$ is a sample complexity as in Theorem 4.8.1. Then, L' obtains i samples $x^1,\dots,x^i$ from the example oracle $\mathsf{EX}_{n,s_0,D}$ and outputs a hypothesis h that takes a random seed r for executing L as input and outputs

$$L(x^1\circ\cdots\circ x^i,\varepsilon,\varepsilon;1^{\langle n,s(n),n,\tau,\delta^{-1},\epsilon^{-1}\rangle};r),$$

where $\tau=O(m_0t(n))$ is specified later. It is easy to verify that L' halts in polynomial time in n,ϵ^{-1}, and δ^{-1}, and the query complexity is $m_0=O((s(n)+\log n)\epsilon^{-2}\delta)$.

We verify that L' learns every $t(n)$-time samplable ACDs (G,D) of $s(n)$-bit initial state. Notice that G determines a distribution family $\mathcal{G}=\{\mathcal{G}_{s(n)}\}_{n\in\mathbb{N}}$, where each $\mathcal{G}_{s(n)}$ is a distribution of $G(1^n)$. Furthermore, D determines a distribution family $\mathcal{D}=\{\mathcal{D}_{n,s_0}\}_{n\in\mathbb{N},s_0\in\{0,1\}^{\le s(n)}}$, where $\mathcal{D}_{n,s_0}$ is a distribution of an *infinitely* long string $x^1x^2x^3\cdots$, where each x^i is a i-th sample generated by $D(1^n,\text{-})$ with initial state s_0. Then, for every $n,\epsilon^{-1},\delta^{-1}\in\mathbb{N}$, the distribution $\mathcal{G}_{s(n)}$ is samplable in $t(n)$ time, and for every initial state $s_0\in\{0,1\}^{\le s}$, the m_0n-bit prefix of $\mathcal{D}_{n,s_0}$ is samplable in $\tau=O(m_0\cdot t(n))$ time. Without loss of generality, we can assume $\tau\ge t(n)$. Then, by Theorem 4.8.1, for every large enough $n\in\mathbb{N}$ and every $\epsilon^{-1},\delta^{-1}$ (note that we choose $a=0$ and $b=n$),

$$\Pr_{s_0,i,x^1,\dots,x^i}\left[\mathsf{L}_1(L(x^1\circ\cdots\circ x^i,\varepsilon,\varepsilon;1^{\langle n,s(n),n,\tau,\delta^{-1},\epsilon^{-1}\rangle}),\mathsf{Label}_i^{s_0,\varepsilon})\le\epsilon\right]$$
$$=\Pr_{s_0,L',x^1,\dots,x^i}\left[\mathsf{L}_1(h(r),D_i^{s_0}(x^1,\dots,x^i))\le\epsilon\right]\ge 1-\delta,$$

where $s_0 \leftarrow G(1^n)$, $i \sim [m_0]$, and $x^1, \ldots, x^i$ are samples generated by $D(1^n, \text{-})$ with initial state s_0. Therefore, L' satisfies the requirements in Definition 4.8.4. □

4.8.3 Universal Top-k Prediction

In this section, we consider a natural task of predicting the next outcome by producing the top k most likely candidates with estimated likelihood for a given $k \in \mathbb{N}$; e.g., {(sunny, 0.8), (cloudy, 0.15), (rainy, 0.02)} in the weather forecast when $k = 3$. We show that this learning task is feasible on average under the non-existence of OWF.

Corollary 4.8.6 (Universal top-k prediction)**.** *If there is no infinitely-often one-way function, then there exist a polynomial-time probabilistic algorithm L and a polynomial m_L such that for every $t_G(s)$-time samplable family $\mathcal{G} = \{\mathcal{G}_s\}_{s\in\mathbb{N}}$ and every $t_D(n,|z|)$-time samplable family $\mathcal{D} = \{\mathcal{D}_{n,z}\}_{n\in\mathbb{N},z\in\{0,1\}^*}$, where each $\mathcal{G}_s$ is over s-bit strings, and each $\mathcal{D}_{n,z}$ is over binary strings, for every large enough $n \in \mathbb{N}$ and for every $n, s, a, b, k, m, t, \epsilon^{-1}, \delta^{-1}, \lambda^{-1} \in \mathbb{N}$ with $t \geq \max\{t_G(s), t_D(n,s)\}$, $k \leq 2^b$, and $m \geq m_L(\log n, s, b, \epsilon^{-1}, \delta^{-1}, \lambda^{-1})$, the following hold with probability at least $1-\delta$ over the choice of $z \sim \mathcal{G}_s, i \sim [m], x^{<i} \sim \mathcal{D}_{n,z}^{<i}$: With probability at least $1-\epsilon$ over the choice of $x^i \sim \mathcal{D}_{n,z}^{i,x^{<i}}$ and the randomness for L, the learner L satisfies the following:*

- $L(x^{<i}, x^i, 1^k; 1^{\langle n,s,b,t,\epsilon^{-1},\delta^{-1},\lambda^{-1}\rangle})$ *outputs* $(y_1,p_1),\ldots,(y_k,p_k) \in \{0,1\}^b \times [0,1]$.
- *Let* $P = \{p_1^*, \ldots, p_{2^b}^*\}$ *be an ordered multi-set defined as* $P = \{\mathsf{Label}_i^{z,x^i}(y) : y \in \{0,1\}^b\}$ *and* $p_j^* \geq p_{j+1}^*$ *for every* $j \in [2^b - 1]$ *(i.e., P is a ranking of probabilities of the next labels). Then, for each* $j \in [k]$,

$$\left|p_j - p_j^*\right| \leq \lambda \text{ and } \left|p_j - \mathsf{Label}_i^{z,x^i}(y_j)\right| \leq \lambda.$$

Furthermore, $m_L(\log n, s, b, \epsilon^{-1}, \delta^{-1}, \lambda^{-1}) = O(\log n + r)b\lambda^{-2}\epsilon^{-3}(\log \epsilon^{-1})\delta^{-1})$.

Proof. We construct a cheating learner $L_{cheat}^{?}$ such that for every $b, k, \lambda^{-1}, \epsilon^{-1} \in \mathbb{N}$ with $k \leq 2^b$ and every distribution Label over $\{0,1\}^b$, the learner $L_{cheat}^{\mathsf{Label}}$ satisfies the following with probability at least $1 - \epsilon/2$ over the choice of samples according to Label:

- $L_{cheat}^{\mathsf{Label}}(1^k, 1^{\langle b,\lambda^{-1},\epsilon^{-1}\rangle})$ outputs $(y_1,p_1),\ldots,(y_k,p_k) \in \{0,1\}^b \times [0,1]$;
- Let $P = \{p_1^*, \ldots, p_{2^b}^*\}$ be an ordered multi-set defined as $P = \{\mathsf{Label}(y) : y \in \{0,1\}^b\}$ and $p_j^* \geq p_{j+1}^*$ for every $j \in [2^b - 1]$. Then, for each $j \in [k]$,

$$\left|p_j - p_j^*\right| \leq \lambda \text{ and } \left|p_j - \mathsf{Label}(y_j)\right| \leq \lambda.$$

Furthermore, the query complexity of L_{cheat} is at most $q = O(\lambda^{-2} b \log \epsilon^{-1})$

If the above holds, then by Theorem 4.8.1, we obtain an algorithm L' that simulates L_{cheat} with UE. The learner L is defined as

$$L(x^{<i}, x^i, 1^k; 1^{\langle n,s,b,t,\epsilon^{-1},\delta^{-1},\lambda^{-1}\rangle}) = L'(x^{<i}, x^i, 1^k, 1^{\langle b,\lambda^{-1},\epsilon^{-1}\rangle}; 1^{\langle n,s,b,t,4\epsilon^{-1}\delta^{-1},4\epsilon^{-1}\rangle}),$$

and the sample complexity function is

$$m_L(\log n, s, b, \epsilon^{-1}, \delta^{-1}, \lambda^{-1}) = O((\log n + r)q\epsilon^{-1}\delta^{-1}\epsilon^{-2}) = O((\log n + r)b\lambda^{-2}\epsilon^{-3}(\log \epsilon^{-1})\delta^{-1}).$$

The correctness of L is verified as follows. By Theorem 4.8.1, for every large enough $n \in \mathbb{N}$ and $r, a, b, k, m, t, \epsilon^{-1}, \delta^{-1}, \lambda^{-1} \in \mathbb{N}$ satisfying the assumptions, with probability at least $1-\delta\epsilon/4$ over the choice of $z \sim \mathcal{G}_s, i \sim [m], x^{<i} \sim \mathcal{D}_{n,z}^{<i}, x^i \sim \mathcal{D}_{n,z}^{i,x^{<i}}$, with probability at least $1 - \epsilon/2 - \epsilon/4 = 1 - 3\epsilon/4$ over the choice of randomness for L', the output $(y_1, p_1), \ldots, (y_k, p_k)$ of L' (i.e., L) satisfies the same property as L_{cheat}. By the simple probabilistic argument (i.e., Markov's inequality and the Union bound) with probability at least $1 - \delta$ over the choice of $z \sim \mathcal{G}_s, i \sim [m], x^{<i} \sim \mathcal{D}_{n,z}^{<i}, x^i \sim \mathcal{D}_{n,z}^{i,x^{<i}}$, and with probability at least $1 - 3\epsilon/4 - \epsilon/4 = 1 - \epsilon$ over the choice of $x^i \sim \mathcal{D}_{n,z}^{i,x^{<i}}$ and randomness for L, the same event occurs.

Therefore, the remaining of the proof is the construction of L_{cheat}, which follows from the standard empirical estimation. On input $1^k, 1^{\langle b, \lambda^{-1}, \epsilon^{-1} \rangle}$, the learner L_{cheat} obtains $q := 8\lambda^{-2} b \ln 4\epsilon^{-1}$ samples $z_1, \ldots, z_q \in \{0,1\}^b$ from Label and counts $m_y := |\{i \in [q] : z^i = y\}|$ for every $y \in \{0,1\}^b$. Let $\tilde{y}_1, \ldots, \tilde{y}_{2^b} \in \{0,1\}^b$ be the ordering of $\{0,1\}^b$ according to the largeness of m_y, i.e., $m_{\tilde{y}_j} \geq m_{\tilde{y}_{j+1}}$ for each $j \in [2^b - 1]$. Then, L_{cheat} outputs k pairs $(\tilde{y}_1, m_{\tilde{y}_1}/q), \ldots, (\tilde{y}_k, m_{\tilde{y}_k}/q)$.

We verify the correctness of L_{cheat}. For each $j \in [2^b]$, let $p_j = m_{\tilde{y}_j}/q$. Notice that $p_1 \geq p_2 \geq \cdots \geq p_{2^b}$. For each $y \in \{0,1\}^b$, by Hoeffding's inequality, it holds that $m_y/q \in [\mathsf{Label}(y) \pm \lambda/4]$ with probability at least $1 - 2e^{-2q(\lambda/4)^2} \geq 1 - e^{-b} \cdot \epsilon/2 \geq 1 - 2^{-b} \cdot \epsilon/2$. By the union bound, every $y \in \{0,1\}^b$ satisfies that $m_y/q \in [\mathsf{Label}(y) \pm \lambda/4]$ with probability at least $1 - \epsilon/2$. We assume that this event occurs. Then, for every $j \in [k]$, it trivially holds that $|p_j - \mathsf{Label}(\tilde{y}_j)| \leq \lambda/4 \leq \lambda$. We also show that $|p_j - p_j^*| \leq \lambda$ (it indeed holds for any $j \in [2^b]$) as follows: Let $y_1^*, \ldots, y_{2^b}^*$ be the ordering of $\{0,1\}^b$ such that $p_j^* = \mathsf{Label}(y_j^*)$ for each j, where we break ties arbitrarily. Let $\ell \in [2^b]$ be an index such that $\tilde{y}_\ell = y_1^*$. Then, for any $j < \ell$, it holds that $\mathsf{Label}(y_1^*) - \mathsf{Label}(\tilde{y}_j) \leq \lambda/2$; otherwise,

$$p_\ell \geq \mathsf{Label}(\tilde{y}_\ell) - \lambda/4 = \mathsf{Label}(y_1^*) - \lambda/4 > \mathsf{Label}(\tilde{y}_j) + \lambda/2 - \lambda/4 \geq (p_j - \lambda/4) + \lambda/4 = p_j.$$

This implies that $|p_j - p_j^*| \leq \lambda$ for every $j \leq \ell$ because (i) there are at least ℓ elements (including y_1^*) whose outcome probability according to Label is at least $\mathsf{Label}(y_1^*) - \lambda/2$; (ii) thus, $p_j^* \geq \mathsf{Label}(y_1^*) - \lambda/2$ for every $j \leq \ell$; and (iii) it holds that, for every $j \leq \ell$,

$$p_j \geq p_\ell \geq \mathsf{Label}(\tilde{y}_\ell) - \lambda/4 = \mathsf{Label}(y_1^*) - \lambda/4 \geq \mathsf{Label}(y_j^*) - \lambda/4 = p_j^* - \lambda/4 \geq p_j^* - \lambda$$

and

$$p_j \leq \mathsf{Label}(\tilde{y}_j) + \lambda/4 \leq \mathsf{Label}(y_1^*) + \lambda/4 \leq p_j^* + \lambda/2 + \lambda/4 \leq p_j^* + \lambda.$$

Next, let $\ell' (> \ell)$ be an index such that $\mathsf{Label}(\tilde{y}_{\ell'}) = \max_{j > \ell} \mathsf{Label}(\tilde{y}_j)$. By the same argument as above, we can show that $|p_j - p_j^*| \leq \lambda$ for every $j \in \mathbb{N}$ with $\ell + 1 \leq j \leq \ell'$. We continue this argument until we run out of all elements and obtain that $|p_j - p_j^*| \leq \lambda$ for every $j \in [2^b]$. □

4.8.4 Universal Likelihood Estimation

In this section, we consider a natural task of estimating the probability that a given label is observed as the next outcome within an additive error; e.g., the probability of "rainy" in the weather forecast. We show that this learning task is also feasible on average under the non-existence of OWF.

Corollary 4.8.7 (Universal likelihood estimation)**.** *If there is no infinitely-often one-way function, then there exist a polynomial-time probabilistic algorithm L and a polynomial m_L such that for every $t_G(s)$-time samplable family $\mathcal{G} = \{\mathcal{G}_s\}_{s \in \mathbb{N}}$ and every $t_D(n, |z|)$-time samplable family $\mathcal{D} = \{\mathcal{D}_{n,z}\}_{n \in \mathbb{N}, z \in \{0,1\}^*}$, where each $\mathcal{G}_s$ is over s-bit strings, and each $\mathcal{D}_{n,z}$ is over binary strings, for*

all large enough $n \in \mathbb{N}$, *all* $s, a, b, k, m, t, \epsilon^{-1}, \delta^{-1}, \lambda^{-1} \in \mathbb{N}$ *with* $t \geq \max\{t_G(s), t_D(n,s)\}$ *and* $m \geq m_L(\log n, s, k, \epsilon^{-1}, \delta^{-1}, \lambda^{-1})$, *and all* $y_1, \ldots, y_k \in \{0,1\}^b$, *the following holds with probability at least* $1-\delta$ *over the choice of* $z \sim \mathcal{G}_s, i \sim [m], x^{<i} \sim \mathcal{D}_{n,z}^{<i}$: *With probability at least* $1-\epsilon$ *over the choice of* $x^i \sim \mathcal{D}_{n,z}^{i,x^{<i}}$ *and randomness for* L, *the learner* L *satisfies the following:*

> $L(x^{<i}, x^i, y_1, \ldots, y_k; 1^{\langle n,s,t,\epsilon^{-1},\delta^{-1},\lambda^{-1}\rangle})$ *outputs* $p_1 \ldots, p_k \in [0,1]$ *satisfying that, for each* $j \in [k]$,
>
> $$\left| p_j - \mathsf{Label}_i^{z,x^i}(y_i) \right| \leq \lambda.$$

Furthermore, $m_L(\log n, s, k, \epsilon^{-1}, \delta^{-1}, \lambda^{-1}) = O((\log n + s)\lambda^{-2}\epsilon^{-3}(\log \epsilon^{-1})\delta^{-1} \log k)$.

Proof. The outline is the same as the proof of Corollary 4.8.6. First, we construct a cheating learner $L_{cheat}^{?}$ such that for every $b, k, \lambda^{-1} \in \mathbb{N}$, every $y_1, \ldots, y_k \in \{0,1\}^b$, and every distribution Label over $\{0,1\}^b$, the learner $L_{cheat}^{\mathsf{Label}}$ satisfies the following with probability at least $1 - \epsilon/2$ over the choice of samples drawn from Label:

> $L_{cheat}^{\mathsf{Label}}(y_1, \ldots, y_k, 1^{\langle \lambda^{-1}, \epsilon^{-1}\rangle})$ outputs $p_1 \ldots, p_k \in [0,1]$ satisfying that $|p_j - \mathsf{Label}(y_i)| \leq \lambda$ for each $j \in [k]$.

Furthermore, the query complexity of L_{cheat} is at most $q = O(\lambda^{-2} \log(k\epsilon^{-1}))$.

Then, by Theorem 4.8.1, we obtain a learner L' that simulates L_{cheat} by UE and construct the learner L is defined as

$$L(x^{<i}, x^i, y_1, \ldots, y_k; 1^{\langle n,s,t,\epsilon^{-1},\delta^{-1},\lambda^{-1}\rangle}) = L'(x^{<i}, x^i, y_1, \ldots, y_k, 1^{\langle \lambda^{-1}, \epsilon^{-1}\rangle}; 1^{\langle n,s,b,t,4\epsilon^{-1}\delta^{-1},4\epsilon^{-1}\rangle}),$$

and the sample complexity function is

$$m_L(\log n, s, k, \epsilon^{-1}, \delta^{-1}, \lambda^{-1}) = O((\log n + s)q\epsilon^{-1}\delta^{-1}\epsilon^{-2}) = O((\log n + s)\lambda^{-2}\epsilon^{-3}(\log \epsilon^{-1})\delta^{-1} \log k).$$

The correctness of L is verified in the same way as Corollary 4.8.6. Thus, we only present the construction of L_{cheat}.

The cheating learner L_{cheat} is constructed based on the standard empirical estimation. On input $y_1, \ldots, y_k$ and $\langle \lambda^{-1}, \epsilon^{-1}\rangle$, the learner L_{cheat} obtains $q := 2^{-1}\lambda^{-2}\ln(4k\epsilon^{-1})$ samples $z_1, \ldots, z_q \in \{0,1\}^b$ from Label and counts $m_j := |\{i \in [q] : z_i = y_j\}|$ for each $j \in [k]$. Then, L_{cheat} outputs $p_j = m_j/q$ for each $j \in [k]$.

The correctness of L_{cheat} is verified as follows. By Hoeffding's inequality, it holds that $m_j/q \in [\mathsf{Label}(y_j) \pm \lambda]$ with probability at least $1 - 2e^{-2q\lambda^2} \geq 1 - \epsilon/(2k)$. By the union bound, it holds that $p_j = m_j/q \in [\mathsf{Label}(y_j) \pm \lambda]$ for all $j \in [k]$ with probability at least $1 - \epsilon/2$. □

4.9 Minimizing Expected Loss by Universal Extrapolation

In this section, we focus on minimizing the expected loss in the online learning framework in Section 4.8. We show that if the value of the loss function is bounded above by $c > 0$, then we can obtain the lower bound of the required number of stages as a function in c instead of the number of queries. As an application, we obtain a universal agnostic learner of better sample complexity than ones obtained directly from Theorem 4.8.1.

As in Section 4.8, we use $a \in \mathbb{N} \cup \{0\}$ and $b, m \in \mathbb{N}$ to refer to the size of advice string, the size of each label, and the total number of stages (i.e., sample complexity), respectively. The learning

framework and notations are also the same as in Section 4.8, and the only differences are the following: (i) a learner is given a past stream $x^{<i}$ and advice information x^i and chooses an action $\alpha_{x^{<i},x_i}$ from a set $\mathcal{A}$ of actions; and (ii) the goal of the learner is to minimize the expected loss with respect to a loss function $l\colon \mathcal{A} \times \{0,1\}^b \to \mathbb{R}_{\geq 0}$, i.e., the learner tries to minimize

$$\mathop{\mathbb{E}}_{x^i,y^i}[l(\alpha_{x^{<i},x_i}, y^i)],$$

where $x^i \sim \mathcal{D}_{n,z}^{i,x^{<i}}$ and $y^i \sim \mathsf{Label}_i^{z,x^i}$.

First, we present the meta-theorem for minimizing the expected loss, which yields better sample complexity when the cheating learner requires polynomially many queries for minimizing the expected loss with respect to a loss function bounded above by a small value.

Definition 4.9.1 (Action set and bounded loss function)**.** *An action set $\mathcal{A} = \{\mathcal{A}_{w,b}\}_{w\in\{0,1\}^*,b\in\mathbb{N}}$ is defined as a family of subsets, where $\mathcal{A}_{w,b} \subseteq \{0,1\}^*$. For a function $c\colon \{0,1\}^* \times \mathbb{N} \to \mathbb{R}_{\geq 0}$, a loss function $l\colon \{0,1\}^* \times \{0,1\}^* \to \mathbb{R}_{\geq 0}$ is said to be c-bounded (with respect to $\mathcal{A}$) if for every $w \in \{0,1\}^*$, every $b \in \mathbb{N}$, every $\alpha \in \mathcal{A}_{w,b}$, and every $y \in \{0,1\}^{\leq b}$, it holds that $l(\alpha, y) \leq c(w, b)$.*

Theorem 4.9.2. *Let $\mathcal{A} = \{\mathcal{A}_{w,b}\}_{w\in\{0,1\}^*,b\in\mathbb{N}}$ be an action set, and let $l\colon \{0,1\}^* \times \{0,1\}^* \to \mathbb{R}_{\geq 0}$ be a c-bounded loss function (with respect to $\mathcal{A}$) for a polynomial-time computable function $c\colon \{0,1\}^* \times \mathbb{N} \to \mathbb{R}_{\geq 0}$.*

Suppose that UE *in Theorem 4.7.1 exists. Then, for every oracle machine (cheating learner) $L^?_{cheat}$ that outputs an action in a set $\mathcal{A}_{w,b}$ with polynomial-time computable query complexity $q(w)$ (where w denotes an input for $L^?_{cheat}$, and b denotes the length of each label), there exist a polynomial m_0 and a randomized algorithm L that outputs an action in the same set $\mathcal{A}_{w,b}$ satisfying the following: for every $t_G(s)$-time samplable family $\mathcal{G} = \{\mathcal{G}_s\}_{s\in\mathbb{N}}$ and every $t_D(n,|z|)$-time samplable family $\mathcal{D} = \{\mathcal{D}_{n,z}\}_{n\in\mathbb{N},z\in\{0,1\}^*}$, where each $\mathcal{G}_s$ is over s-bit strings, and each $\mathcal{D}_{n,z}$ is over binary strings, there exists $n_0 \in \mathbb{N}$ such that for every $n, s, a, b, t, \epsilon^{-1}, \delta^{-1} \in \mathbb{N}$ with $n \geq n_0$ and $t \geq \max\{t_G(s), t_D(n,s)\}$, every auxiliary input $w \in \{0,1\}^*$, and every $m \geq m_0(\log n, s, c(w,b), \epsilon^{-1}, \delta^{-1})$,*

$$\Pr_{z,i,x^{<i}}\left[\mathop{\mathbb{E}}_{x^i,y^i,L}\left[l(L(x^{<i}, x^i, w; 1^{\langle n,r,b,t,\epsilon^{-1},\delta^{-1}\rangle}), y^i)\right] \leq \mathop{\mathbb{E}}_{x^i}\left[\min_{\alpha\in\mathcal{A}_{w,b}} \mathop{\mathbb{E}}_{y^i}[l(\alpha, y^i)]\right] + 2\Delta_{L_{cheat}}(w,b) + \epsilon\right] \geq 1-\delta,$$

where $z \sim \mathcal{G}_r, i \sim [m], x^{<i} \sim \mathcal{D}_{n,z}^{<i}, x^i \sim \mathcal{D}_{n,z}^{i,x^{<i}}$, $y^i \sim \mathsf{Label}_i^{z,x^i}$, and

$$\Delta_{L_{cheat}}(w,b) := \sup_{\mathcal{O}:\text{distribution over } \{0,1\}^{\leq b}} \left(\mathop{\mathbb{E}}_{\mathcal{O},y\sim\mathcal{O}}[l(L^{\mathcal{O}}_{cheat}(w), y)] - \min_{\alpha\in\mathcal{A}_{w,b}} \mathop{\mathbb{E}}_{y\sim\mathcal{O}}[l(\alpha, y)]\right).$$

Furthermore, $m_0(\log n, s, c, \epsilon^{-1}, \delta^{-1}) = O((\log n + s)\cdot c^2 \cdot \epsilon^{-2}\delta^{-2})$, $n_0 = n'(d(\mathcal{G}), d(\mathcal{D}))$ for a universal function $n'\colon \mathbb{N}\times\mathbb{N} \to \mathbb{N}$, and L halts in polynomial time in the input length and the running time of L_{cheat}.

The proof is given in Section 4.9.1. As an application, we show that average-case universal agnostic learning with optimal sample complexity (for the 0-1 loss) is feasible under the non-existence of one-way functions in Section 4.9.2. In Section 4.9.3, we consider agnostic learning for general loss functions in the case in which the number of labels is polynomially bounded.

4.9.1 A Proof of Meta-Theorem

First, we show the following key lemma.

Lemma 4.9.3. *Let $b \in \mathbb{N}$. Let $\mathcal{A} \subseteq \{0,1\}^*$, and let $l\colon \mathcal{A} \times \{0,1\}^{\leq b} \to \mathbb{R}_{\geq 0}$ be a loss function satisfying that there exists $C > 0$ such that $l(\alpha, y) \leq C$ for every $\alpha \in \mathcal{A}$ and $y \in \{0,1\}^{\leq b}$.*

For every distribution $\mathcal{D}$ on $\{0,1\}^*$ such that $\mathcal{D}$ has a t_D-time sampler described by d bits, and for every $t, a, b, m \in \mathbb{N}$ with $t \geq \tau_{\mathrm{dom}}(d, t_D)$,

$$\mathop{\mathbb{E}}_{i, x^{<i}, x^i} \left[\sum_{y \in \{0,1\}^{\leq b}} \left| \mathsf{Next}_b(\mathcal{D}, x^{<i}x^i)(y) - \mathsf{Next}_b(\mathrm{Q}^t, x^{<i}x^i)(y) \right| \cdot \max_{\alpha \in \mathcal{A}} l(\alpha, y) \right] \leq C \cdot \sqrt{\frac{O(d)}{m}},$$

and for every oracle machine $I^?$ that outputs a string in $\mathcal{A}$,

$$\mathop{\mathbb{E}}_{i, x^{<i}, x^i} \left[\left| \mathop{\mathbb{E}}_{y \sim \mathsf{Next}_b(\mathrm{Q}^t, x^{<i}x^i)} [l(I^{\mathsf{Next}_b(\mathcal{D}, x^{<i}x^i)}, y)] - \mathop{\mathbb{E}}_{y \sim \mathsf{Next}_b(\mathcal{D}, x^{<i}x^i)} [l(I^{\mathsf{Next}_b(\mathcal{D}, x^{<i}x^i)}, y)] \right| \right] \leq C \cdot \sqrt{\frac{O(d)}{m}},$$

where $i \sim [m], x^{<i} \sim \mathcal{D}^{<i}, x^i \sim \mathcal{D}^{i, x^{<i}}$, and the hidden constant in $O(d)$ depends on only the universal Turing machine.

Proof. The first claim is verified as follows:

$$\begin{aligned}
&\mathop{\mathbb{E}}_{i, x^{<i}, x^i} \left[\sum_{y \in \{0,1\}^{\leq b}} \left| \mathsf{Next}_b(\mathcal{D}, x^{<i}x^i)(y) - \mathsf{Next}_b(\mathrm{Q}^t, x^{<i}x^i)(y) \right| \cdot \max_{\alpha \in \mathcal{A}} l(\alpha, y) \right] \\
&\leq C \cdot \mathop{\mathbb{E}}_{i, x^{<i}, x^i} \left[\sum_{y \in \{0,1\}^{\leq b}} \left| \mathsf{Next}_b(\mathcal{D}, x^{<i}x^i)(y) - \mathsf{Next}_b(\mathrm{Q}^t, x^{<i}x^i)(y) \right| \right] \\
&= 2C \cdot \mathop{\mathbb{E}}_{i, x^{<i}, x^i} \left[\mathsf{L}_1 \left(\mathsf{Next}_b(\mathcal{D}, x^{<i}x^i), \mathsf{Next}_b(\mathrm{Q}^t, x^{<i}x^i) \right) \right] \\
&\leq 2C \cdot \mathop{\mathbb{E}}_{i, x^{<i}, x^i} \left[\sqrt{2^{-1} \cdot \mathrm{KL} \left(\mathsf{Next}_b(\mathcal{D}, x^{<i}x^i) || \mathsf{Next}_b(\mathrm{Q}^t, x^{<i}x^i) \right)} \right] \\
&\leq \sqrt{2}C \cdot \sqrt{\mathop{\mathbb{E}}_{i, x^{<i}, x^i} \left[\mathrm{KL} \left(\mathsf{Next}_b(\mathcal{D}, x^{<i}x^i) || \mathsf{Next}_b(\mathrm{Q}^t, x^{<i}x^i) \right) \right]} \\
&\leq \sqrt{2}C \cdot \sqrt{\frac{O(d)}{m}} = C \cdot \sqrt{\frac{O(d)}{m}},
\end{aligned}$$

where the first inequality holds by $l(\alpha, y) \leq C$ for every $\alpha \in \mathcal{A}$ and $y \in \{0,1\}^{\leq b}$, the second inequality follows from Pinsker's inequality (Fact 4.5.1), the third inequality follows from Jensen's inequality, the last inequality follows from Lemma 4.8.2 for a trivial 1-query algorithm I that outputs a sample obtained from the oracle.

The second claim is also verified as follows.

$$\begin{aligned}
&\mathop{\mathbb{E}}_{i,x^{<i},x^i}\left[\left|\mathop{\mathbb{E}}_{y\sim \mathsf{Next}_b(\mathrm{Q}^t,x^{<i}x^i)}[l(I^{\mathsf{Next}_b(\mathcal{D},x^{<i}x^i)},y)]-\mathop{\mathbb{E}}_{y\sim \mathsf{Next}_b(\mathcal{D},x^{<i}x^i)}[l(I^{\mathsf{Next}_b(\mathcal{D},x^{<i}x^i)},y)]\right|\right]\\
&=\mathop{\mathbb{E}}_{i,x^{<i},x^i}\left[\left|\sum_{y\in\{0,1\}^{\le b}}(\mathsf{Next}_b(\mathrm{Q}^t,x^{<i}x^i)(y)-\mathsf{Next}_b(\mathcal{D},x^{<i}x^i)(y))\,\mathbb{E}[l(I^{\mathsf{Next}_b(\mathcal{D},x^{<i}x^i)},y)]\right|\right]\\
&\le\mathop{\mathbb{E}}_{i,x^{<i},x^i}\left[\sum_{y\in\{0,1\}^{\le b}}\left|\mathsf{Next}_b(\mathrm{Q}^t,x^{<i}x^i)(y)-\mathsf{Next}_b(\mathcal{D},x^{<i}x^i)(y)\right|\cdot\max_{\alpha\in\mathcal{A}}l(\alpha,y)\right]\\
&\le C\cdot\sqrt{\frac{O(d)}{m}},
\end{aligned}$$

where the last inequality follows from the first claim. □

Now, we derive Theorem 4.9.2 from Theorem 4.7.1 and Lemma 4.9.3, which is a time-bounded variant of the theory of universal prediction in [MF98].

Proof of Theorem 4.9.2. Let UE be the universal extrapolation algorithm in Theorem 4.7.1. We use the same notations n_{UE} and τ_{UE} as in Theorem 4.7.1.

For every $t_G(s)$-time samplable distribution family $\mathcal{G}$ and every $t_D(n,|z|)$-time samplable distribution family $\mathcal{D}$, we define another distribution family $\mathcal{D}'=\{\mathcal{D}'_{\langle n,s\rangle}\}_{n,s\in\mathbb{N}}$ as $\mathcal{D}'_{\langle n,s\rangle}\equiv\mathcal{D}_{n,\mathcal{G}_s}$. It holds that $d(\mathcal{D}')=O(d(\mathcal{G})+d(\mathcal{D}))$. Let $n_1\colon\mathbb{N}\times\mathbb{N}\to\mathbb{N}$ be a function such that $n_1(d(\mathcal{G}),d(\mathcal{D}))=n_{\mathsf{UE}}(d(\mathcal{D}'))=2^{O(d(\mathcal{G})+d(\mathcal{D}))}$. Furthermore, we define a polynomial $\tau_0(t,t')$ such that for every $n,s\in\mathbb{N}$, the distribution $\mathcal{D}_{\langle n,s\rangle}$ is samplable in time $\tau_0(t_G(s),t_D(n,s))$.

As the proof of Theorem 4.8.1, we construct L that executes L_{cheat}, where the query access to Label is simulated by UE. On input $x^{<i},x^i,w,1^{\langle n,s,b,t,\epsilon^{-1},\delta^{-1}\rangle}$, the learner L executes the cheating learner $L^{?}_{cheat}(w)$, where L answers each query to Label_i^{z,x^i} by

$$\mathsf{ans}\leftarrow\mathsf{UE}(x^{<i}x^i;1^{\langle n,s,b,t',\epsilon'^{-1},8C\epsilon^{-1}\delta^{-1}\rangle})$$

for $t'=\max\{\tau_{\mathsf{UE}}(\langle n,s\rangle,\tau_0(t,t)),\tau_1(n,s,t)\}$, $C:=c(w,b)$, and $\epsilon'=\epsilon/(4Cq(w))$ (with fresh randomness for each access), where τ_1 is a polynomial specified later. It is easy to verify that L halts in polynomial time in the input length and the running time of L_{cheat}. Below, we show the correctness of L. For readability, we omit the parameters for L and UE.

For the correctness, we evaluate the following expectation for $z\sim\mathcal{G}_s,i\sim[m],x^{<i}\sim\mathcal{D}^{<i}_{n,z}$:

$$\mathop{\mathbb{E}}_{x^i,y^i,L}\left[l(L(x^{<i},x^i,w),y^i)\right],$$

where $x^i\sim\mathcal{D}^{i,x^{<i}}_{n,z}$, $y^i\sim\mathsf{Label}_i^{z,x^i}$.

For every $z, i, x^{<i}$, we have

$$\begin{aligned}
&\mathop{\mathbb{E}}_{x^i,y^i,L}\left[l(L(x^{<i},x^i,w),y^i)\right] \\
&= \mathop{\mathbb{E}}_{x^i,L}\left[\sum_{y^i\in\{0,1\}^{\le b}} \mathsf{Next}_b(\mathcal{D}_{n,z},x^{<i}x^i)(y^i)\cdot l(L(x^{<i},x^i,w),y^i)\right] \\
&\le \mathop{\mathbb{E}}_{x^i,L}\left[\sum_{y^i\in\{0,1\}^{\le b}} (\mathsf{Next}_b(\mathrm{Q}^t,x^{<i}x^i)(y)+|\mathsf{Next}_b(\mathcal{D}_{n,z},x^{<i}x^i)(y^i)-\mathsf{Next}_b(\mathrm{Q}^t,x^{<i}x^i)(y)|)\cdot l(L(x^{<i},x^i,w),y^i)\right] \\
&= S_1+S_2,
\end{aligned}$$

where

$$\begin{aligned}
S_1 &:= \mathop{\mathbb{E}}_{x^i,L}\left[\sum_{y^i\in\{0,1\}^{\le b}} \mathsf{Next}_b(\mathrm{Q}^t,x^{<i}x^i)(y^i)\cdot l(L(x^{<i},x^i,w),y^i)\right] \\
&= \mathop{\mathbb{E}}_{x^i,L}\left[\mathop{\mathbb{E}}_{y^i\sim\mathsf{Next}_b(\mathrm{Q}^t,x^{<i}x^i)}[l(L(x^{<i},x^i,w),y^i)]\right] \\
S_2 &:= \mathop{\mathbb{E}}_{x^i,L}\left[\sum_{y^i\in\{0,1\}^{\le b}} |\mathsf{Next}_b(\mathcal{D}_{n,z},x^{<i}x^i)(y^i)-\mathsf{Next}_b(\mathrm{Q}^t,x^{<i}x^i)(y^i)|\cdot l(L(x^{<i},x^i,w),y^i)\right] \\
&\le \mathop{\mathbb{E}}_{x^i}\left[\sum_{y^i\in\{0,1\}^{\le b}} |\mathsf{Next}_b(\mathcal{D}_{n,z},x^{<i}x^i)(y^i)-\mathsf{Next}_b(\mathrm{Q}^t,x^{<i}x^i)(y^i)|\cdot \max_{\alpha\in\mathcal{A}_{w,b}} l(\alpha,y^i)\right].
\end{aligned}$$

First, we show the upper bound on S_2. Let D be the sampler for $\mathcal{D}$ described by $d(\mathcal{D})$ bits. We choose a sufficiently large constant $c\ge 1$ so that (i) for every $d,d'\in\mathbb{N}$, $n_1(d,d')\le 2^{c(d+d')}$, and (ii) for every $n,s\in\mathbb{N}$ and $z\in\{0,1\}^s$, the sampler $D(1^n,z;\text{-})$ is described by $c(d(\mathcal{D})+\log n+s)$ bits. Let $n'(d,d'):=2^{c(d+d')}\ge n_1(d,d')$ and $n_0:=n'(d(\mathcal{G}),d(\mathcal{D}))$.

By Lemma 4.9.3, there exists a polynomial τ_1 such that for every $n,s,t,m,a,b\in\mathbb{N}$, $z\in\{0,1\}^s$, and $w\in\{0,1\}^*$ with $n\ge n_0$ and $t\ge\tau_1(n,s,t_D(n,s))$,

$$\begin{aligned}
\mathop{\mathbb{E}}_{i,x^{<i}}[S_2] &\le \mathop{\mathbb{E}}_{i,x^{<i},x^i}\left[\sum_{y^i\in\{0,1\}^{\le b}} |\mathsf{Next}_b(\mathcal{D}_{n,z},x^{<i}x^i)(y^i)-\mathsf{Next}_b(\mathrm{Q}^t,x^{<i}x^i)(y^i)|\cdot \max_{\alpha\in\mathcal{A}_{w,b}} l(\alpha,y^i)\right] \\
&\le C\cdot\sqrt{\frac{O(d(\mathcal{D})+\log n+s)}{m}} \\
&\le \sqrt{\frac{c'C^2(s+\log n)}{m}},
\end{aligned}$$

for some universal constant $c'>0$, where we use the fact that $\log n\ge\log n_0\ge d(\mathcal{D})$.

Let $m_0:=\frac{256c'C^2(s+\log n)}{\epsilon^2\delta^2}=O(\frac{C^2\cdot(s+\log n)}{\epsilon^2\delta^2})$. Then, for every $m\ge m_0$, we have

$$\mathop{\mathbb{E}}_{i,x^{<i}}[S_2]\le\sqrt{\frac{c'C^2(s+\log n)}{m}}\le\sqrt{\frac{c'C^2(s+\log n)}{m_0}}=\frac{\epsilon\delta}{16}.$$

It is easy to verify that S_2 is always non-negative. Thus, by Markov's inequality,

$$\Pr_{i,x^{<i}}\left[S_2 \le \frac{\epsilon}{4}\right] \ge 1 - \frac{\delta}{4}. \tag{4.6}$$

We remark that the above holds for any $n, s, t, m, \epsilon^{-1}, \delta^{-1}, a, b \in \mathbb{N}$, any $z \in \{0,1\}^s$, and any $w \in \{0,1\}^*$ satisfying that $n \ge n_0$, $t \ge \tau_1(n, s, t_D(n, s))$, and $m \ge m_0$.

Next, we show the upper bound on S_1. For readability, we omit "$, x^{<i}x^i$" from $\mathsf{Next}_b(\mathrm{Q}^{t'}, x^{<i}x^i)$ and $\mathsf{Next}_b(\mathcal{D}, x^{<i}x^i)$ and write them as $\mathsf{Next}_b(\mathrm{Q}^{t'})$ and $\mathsf{Next}_b(\mathcal{D})$, respectively.

For each $x^i \in \mathrm{supp}(\mathcal{D}_{n,z}^{i,x^{<i}})$, we define E_{x^i} and S'_{1,x^i} as follows:

$$E_{x^i} := \mathop{\mathbb{E}}_{L, y^i \sim \mathsf{Next}_b(\mathrm{Q}^{t'})}[l(L(x^{<i}, x^i, w), y^i) - l(L_{cheat}^{\mathsf{Next}_b(\mathrm{Q}^{t'})}(w), y^i)]$$

$$S'_{1,x^i} := \left| \mathop{\mathbb{E}}_{y^i \sim \mathsf{Next}_b(\mathrm{Q}^{t'})}[l(L_{cheat}^{\mathsf{Next}_b(\mathcal{D}_{n,z})}(w), y^i)] - \mathop{\mathbb{E}}_{y^i \sim \mathsf{Next}_b(\mathcal{D}_{n,z})}[l(L_{cheat}^{\mathsf{Next}_b(\mathcal{D}_{n,z})}(w), y^i)] \right|.$$

Then, we have

$$\begin{aligned}
S_1 &= \mathop{\mathbb{E}}_{x^i, L, y^i \sim \mathsf{Next}_b(\mathrm{Q}^{t'})}[l(L(x^{<i}, x^i, w), y^i)] \\
&= \mathop{\mathbb{E}}_{x^i, y^i \sim \mathsf{Next}_b(\mathrm{Q}^{t'})}[l(L_{cheat}^{\mathsf{Next}_b(\mathrm{Q}^{t'})}(w), y^i)] + \mathop{\mathbb{E}}_{x^i}[E_{x^i}] \\
&\le \mathop{\mathbb{E}}_{x^i}\left[\min_{\alpha \in \mathcal{A}_{w,b}} \mathop{\mathbb{E}}_{y^i \sim \mathsf{Next}_b(\mathrm{Q}^{t'})}[l(\alpha, y^i)]\right] + \Delta_{L_{cheat}}(w, b) + \mathop{\mathbb{E}}_{x^i}[E_{x^i}] \\
&\le \mathop{\mathbb{E}}_{x^i}\left[\mathop{\mathbb{E}}_{y^i \sim \mathsf{Next}_b(\mathrm{Q}^{t'})}[l(L_{cheat}^{\mathsf{Next}_b(\mathcal{D}_{n,z})}(w), y^i)]\right] + \Delta_{L_{cheat}}(w, b) + \mathop{\mathbb{E}}_{x^i}[E_{x^i}] \\
&\le \mathop{\mathbb{E}}_{x^i}\left[S'_{1,x^i}\right] + \mathop{\mathbb{E}}_{x^i}\left[\mathop{\mathbb{E}}_{y^i \sim \mathsf{Next}_b(\mathcal{D}_{n,z})}[l(L_{cheat}^{\mathsf{Next}_b(\mathcal{D}_{n,z})}(w), y^i)]\right] + \Delta_{L_{cheat}}(w, b) + \mathop{\mathbb{E}}_{x^i}[E_{x^i}] \\
&\le \mathop{\mathbb{E}}_{x^i}\left[S'_{1,x^i}\right] + \mathop{\mathbb{E}}_{x^i}\left[\min_{\alpha \in \mathcal{A}_{w,b}} \mathop{\mathbb{E}}_{y^i \sim \mathsf{Next}_b(\mathcal{D}_{n,z})}[l(\alpha, y^i)]\right] + 2\Delta_{L_{cheat}}(w, b) + \mathop{\mathbb{E}}_{x^i}[E_{x^i}] \\
&= \mathop{\mathbb{E}}_{x^i}\left[\min_{\alpha \in \mathcal{A}_{w,b}} \mathop{\mathbb{E}}_{y^i \sim \mathsf{Label}_i^{z,x^i}}[l(\alpha, y^i)]\right] + 2\Delta_{L_{cheat}}(w, b) + \mathop{\mathbb{E}}_{x^i}\left[S'_{1,x^i}\right] + \mathop{\mathbb{E}}_{x^i}[E_{x^i}],
\end{aligned}$$

where the first and last inequalities follow from the definition of $\Delta_{L_{cheat}}(w, b)$.

For all $n, s, m, t, \epsilon^{-1}, \delta^{-1}, a, b \in \mathbb{N}$, $i \in [m]$, and $w \in \{0,1\}^*$ with $n \ge n_0$, $m \ge m_0$, and $t \ge \max\{t_G(s), t_D(n, s)\}$, we have that $t' \ge \tau_{\mathsf{UE}}(\langle n, s\rangle, \tau_0(t, t)) \ge \tau_{\mathsf{UE}}(\langle n, s\rangle, \tau_0(t_G(s), t_D(n, s)))$, and by Theorem 4.7.1,

$$\Pr_{z \sim \mathcal{G}_s, x^{<i} \sim \mathcal{D}_{n,z}^{<i}, x^i \sim \mathcal{D}_{n,z}^{i,x^{<i}}}\left[\mathsf{L}_1\left(\mathsf{UE}(x^{<i}x^i; 1^{\langle n,s,b,t',\epsilon'^{-1},8C\epsilon^{-1}\delta^{-1}\rangle}), \mathsf{Next}_b(\mathrm{Q}^{t'}, x^{<i}x^i)\right) \le \frac{\epsilon}{4Cq(w)}\right] \ge 1 - \frac{\epsilon\delta}{8C}.$$

By Markov's inequality,

$$\Pr_{z, x^{<i}}\left[\Pr_{x^i}\left[\mathsf{L}_1\left(\mathsf{UE}(x^{<i}x^i), \mathsf{Next}_b(\mathrm{Q}^{t'})\right) \le \frac{\epsilon}{4Cq(w)}\right] \ge 1 - \frac{\epsilon}{4C}\right] \ge 1 - \frac{\delta}{2} \tag{4.7}$$

Remember that (i) L simulates the oracle access by L_{cheat} by $\mathsf{UE}(x^{<i}x^i)$; and (ii) L_{cheat} makes at most $q(w)$ times. Thus, for every $w \in \{0,1\}^*$ and every $z, x^{<i}, x^i$ satisfying the event in inequality (4.7), it holds that

$$\mathsf{L}_1(L(x^{<i}, x^i, w), L_{cheat}^{\mathsf{Next}_b(\mathrm{Q}^{t'})}(w)) = \mathsf{L}_1(L_{cheat}^{\mathsf{UE}(x^{<i}x^i)}(w), L_{cheat}^{\mathsf{Next}_b(\mathrm{Q}^{t'})}(w)) \leq q(w) \cdot \frac{\epsilon}{4Cq(w)} = \frac{\epsilon}{4C}$$

and

$$\begin{aligned}
E_{x^i} &= \mathop{\mathbb{E}}_{L, y^i \sim \mathsf{Next}_b(\mathrm{Q}^{t'})}[l(L(x^{<i}, x^i, w), y^i) - l(L_{cheat}^{\mathsf{Next}_b(\mathrm{Q}^{t'})}(w), y^i)] \\
&\leq \max_{y \in \{0,1\}^{\leq b}} \mathop{\mathbb{E}}_{L, \mathsf{Next}_b(\mathrm{Q}^{t'})}[l(L(x^{<i}, x^i, w), y) - l(L_{cheat}^{\mathsf{Next}_b(\mathrm{Q}^{t'})}(w), y)] \\
&\leq \max_{y \in \{0,1\}^{\leq b}} \sum_{\alpha \in \mathcal{A}_{w,b}} \left(\Pr[L(x^{<i}, x^i, w) = \alpha] - \Pr[L_{cheat}^{\mathsf{Next}_b(\mathrm{Q}^{t'})}(w) = \alpha] \right) \cdot l(\alpha, y) \\
&\leq \max_{\alpha' \in \mathcal{A}_{w,b}, y \in \{0,1\}^{\leq b}} l(\alpha', y) \cdot \sum_{\alpha \in \mathcal{A}_{w,b}} \left(\Pr[L(x^{<i}, x^i, w) = \alpha] - \Pr[L_{cheat}^{\mathsf{Next}_b(\mathrm{Q}^{t'})}(w) = \alpha] \right) \\
&\leq C \cdot \frac{\epsilon}{4C} = \frac{\epsilon}{4}.
\end{aligned}$$

Thus, we have that, for every $z, x^{<i}$ satisfying the event in inequality (4.7),

$$\begin{aligned}
\mathop{\mathbb{E}}_{x^i}[E_{x^i}] &= \mathop{\mathbb{E}}_{x^i, L, y^i \sim \mathsf{Next}_b(\mathrm{Q}^{t'})}[l(L(x^{<i}, x^i, w), y^i) - l(L_{cheat}^{\mathsf{Next}_b(\mathrm{Q}^{t'})}(w), y^i)] \\
&\leq 1 \cdot \frac{\epsilon}{4} + \frac{\epsilon}{4C} \max_{\alpha, \alpha' \in \mathcal{A}_{w,b}, y \in \{0,1\}^{\leq b}} \left(l(\alpha, y) - l(\alpha', y) \right) \\
&\leq \frac{\epsilon}{4} + \frac{\epsilon}{4C} \cdot C = \frac{\epsilon}{2}.
\end{aligned}$$

We also evaluate S'_{1,x^i} in the same way as S_2. By Lemma 4.9.3, we have that, for every $n, s, t, m, a, b \in \mathbb{N}$, $z \in \{0,1\}^s$, and $w \in \{0,1\}^*$ with $n \geq n_0$, $t \geq \tau_1(n, s, t_D(n, s))$, and $m \geq m_0$,

$$\mathop{\mathbb{E}}_{i, x^{<i}}[\mathop{\mathbb{E}}_{x^i}[S'_{1,x^i}]] \leq \sqrt{\frac{c' C^2 (s + \log n)}{m}} \leq \sqrt{\frac{c' C^2 (s + \log n)}{m_0}} = \frac{\epsilon\delta}{16}.$$

Since S'_{1,x^i} is always non-negative, by Markov's inequality,

$$\Pr_{i, x^{<i}}\left[\mathop{\mathbb{E}}_{x^i}[S'_{1,x^i}] \leq \frac{\epsilon}{4}\right] \geq 1 - \frac{\delta}{4}. \tag{4.8}$$

Since $t' \geq \tau_1(n, s, t) \geq \tau_1(n, s, t_D(n, s))$, by inequalities (4.6), (4.7), and (4.8) and the union bound, for every $w \in \{0,1\}^*$, it holds that (i) $\mathbb{E}_{x^i}[E_{x^i}] \leq \epsilon/2$, (ii) $\mathbb{E}_{x^i}[S'_{1,x^i}] \leq \epsilon/4$, and (iii) $S_2 \leq \epsilon/4$ with probability at least $1 - \delta$ over the choice of $z \sim \mathcal{G}_s$, $i \sim [m]$, and $x^{<i} \sim \mathcal{D}_{n,z}^{<i}$. In this

case, we have

$$\begin{aligned}
\mathop{\mathbb{E}}_{x^i,y^i,L}\left[l(L(x^{<i},x^i,w),y^i)\right] &\le S_1+S_2\\
&\le \mathop{\mathbb{E}}_{x^i}\left[\min_{\alpha\in\mathcal{A}_{w,b}}\mathop{\mathbb{E}}_{y^i\sim\mathsf{Label}_i^{z,x^i}}[l(\alpha,y^i)]\right]+2\Delta_{L_{cheat}}(w,b)+\mathop{\mathbb{E}}_{x^i}\left[S'_{1,x^i}\right]+\mathop{\mathbb{E}}_{x^i}[E_{x^i}]+S_2\\
&\le \mathop{\mathbb{E}}_{x^i}\left[\min_{\alpha\in\mathcal{A}_{w,b}}\mathop{\mathbb{E}}_{y^i\sim\mathsf{Label}_i^{z,x^i}}[l(\alpha,y^i)]\right]+2\Delta_{L_{cheat}}(w,b)+\frac{\epsilon}{2}+\frac{\epsilon}{4}+\frac{\epsilon}{4}\\
&= \mathop{\mathbb{E}}_{x^i}\left[\min_{\alpha\in\mathcal{A}_{w,b}}\mathop{\mathbb{E}}_{y^i\sim\mathsf{Label}_i^{z,x^i}}[l(\alpha,y^i)]\right]+2\Delta_{L_{cheat}}(w,b)+\epsilon.
\end{aligned}$$

Hence, we conclude that

$$\Pr_{z,i,x^{<i}}\left[\mathop{\mathbb{E}}_{x^i,y^i,L}\left[l(L(x^{<i},x^i,w),y^i)\right]\le\mathop{\mathbb{E}}_{x^i}\left[\min_{\alpha\in\mathcal{A}_{w,b}}\mathop{\mathbb{E}}_{y^i\sim\mathsf{Label}_i^{z,x^i}}[l(\alpha,y^i)]\right]+2\Delta_{L_{cheat}}(w,b)+\epsilon\right]\ge 1-\delta.$$

□

4.9.2 Universal Agnostic Learning

We introduce the average-case variant of agnostic learning and show that universal average-case agnostic learning is feasible under the non-existence of one-way functions.

In this section, we define a sampler of example size n as a multi-output circuit that outputs a pair $(x,y)\in\{0,1\}^n\times\{0,1\}^{\mathsf{poly}(n)}$ (we call x an example and y a label of x). For any sampler $S\colon\{0,1\}^\ell\to\{0,1\}^n\times\{0,1\}^{\mathsf{poly}(n)}$, we use the notation $S^{(1)}$ (resp. $S^{(2)}$) to refer to the circuit that produces the first (resp. second) half element of S, i.e., $S(r)=(S^{(1)}(r),S^{(2)}(r))$ for each seed $r\in\{0,1\}^\ell$. For convenience, we may identify a sampler $S\colon\{0,1\}^\ell\to\{0,1\}^n\times\{0,1\}^{\mathsf{poly}(n)}$ with a distribution of $S(r)$, where $r\sim\{0,1\}^\ell$. For each sampler S, we define an example oracle EX_S as the oracle that returns $(x,y)\sim S$ for each access. For simplicity, we define the time complexity of sampler as a function in the example size n instead of the seed length ℓ. For any $t,s\in\mathbb{N}$, we say that a sampler S of example size n is t/s-time computable if there exists a program $\Pi_S\in\{0,1\}^{\le s}$ such that $U^t(\Pi_S,r)=S(r_{[\ell]})$ for each seed $r\in\{0,1\}^t$.

In the original agnostic learning model in [KSS94], a learner for a concept class $\mathscr{C}$ is given access to EX_S for an unknown sampler S, and the task is to approximate the best function in $\mathscr{C}$ that approximates the label under S (more generally, the best function in $\mathscr{C}$ that minimizes the expected loss for some loss function) for all samplers S (i.e., in the worst case with respect to S).

To introduce the average-case variant of agnostic learning, we define a distribution on samplers as a family $\mathcal{D}=\{\mathcal{D}_n\}_{n\in\mathbb{N}}$ of distributions, where $\mathcal{D}_n$ is a distribution on descriptions of a sampler of example size n. Note that, for every distribution $\mathcal{D}$ on samplers and every $n\in\mathbb{N}$, we use the notation $\mathcal{D}_n$ to the n-th distribution in $\mathcal{D}$, i.e., the distribution on descriptions of samplers of example size n. Then, our learning model is formulated as follows.

Definition 4.9.4 (Agnostic learning on average)**.** *Let $b\colon\mathbb{N}\to\mathbb{N}$ be a size of each label. Let $\mathscr{C}$ be a concept class defined as a subset of $\{f\colon\{0,1\}^n\to\{0,1\}^{b(n)}:n\in\mathbb{N}\}$ and $\mathcal{D}$ be a distribution on samplers S over $\{0,1\}^n\times\{0,1\}^{b(n)}$ for the example size n. Let $l\colon\{0,1\}^*\times\{0,1\}^*\to\mathbb{R}_{\ge 0}$ be a loss function.*

We say that a randomized oracle L, referred to as an agnostic learner, agnostically learns $\mathscr{C}$ on average under $\mathcal{D}$ for a loss function l if for every sufficiently large $n \in \mathbb{N}$, every $\epsilon^{-1}, \delta^{-1} \in \mathbb{N}$, the learner $L(1^n, 1^{\epsilon^{-1}}, 1^{\delta^{-1}})$ is given access to EX_S, where S is an unknown target sampler over $\{0,1\}^n \times \{0,1\}^{b(n)}$ selected according to $\mathcal{D}_n$, and outputs a circuit $h\colon \{0,1\}^n \to \{0,1\}^{b(n)}$ such that

$$\mathop{\mathbb{E}}_{(x,y)\sim S}[l(h(x),y)] \leq \mathsf{opt}_{\mathscr{C}}(S) + \epsilon,$$

where

$$\mathsf{opt}_{\mathscr{C}}(S) = \min_{f\in\mathscr{C}} \mathop{\mathbb{E}}_{(x,y)\sim S}[l(f(x),y)]$$

with probability at least $1-\delta$ over the choice of $S \sim \mathcal{D}_n$, EX_S, and randomness for L. We define a sample complexity $m(n,\epsilon,\delta)$ of L as the upper bound on the number of query access by $L(1^n, 1^{\epsilon^{-1}}, 1^{\delta^{-1}})$ for each $n, \epsilon^{-1}, \delta^{-1} \in \mathbb{N}$.

Let $\mathscr{D}$ be a class of distributions on samplers. We say that $\mathscr{C}$ is agnostic learnable in polynomial time on average under $\mathscr{D}$ for a loss function l if there exists a polynomial-time agnostic learner that agnostically learns $\mathscr{C}$ on average under $\mathcal{D}$ for every (unknown) $\mathcal{D} \in \mathscr{D}$.

If we do not specify the loss function, we always assume the 0-1 loss function l defined as

$$l(\tilde{y}, y) = \begin{cases} 1 & \text{if } \tilde{y} \neq y \\ 0 & \text{if } \tilde{y} = y. \end{cases}$$

In this case, the requirement for the hypothesis h in Definition 4.9.4 is simply written as follows:

$$\Pr_{(x,y)\sim S}[h(x) \neq y] \leq \mathsf{opt}_{\mathscr{C}}(S) + \epsilon = \min_{f\in\mathscr{C}} \Pr_{(x,y)\sim S}[f(x) \neq y] + \epsilon.$$

Now, we show the following learnability result on universal agnostic learning from Theorem 4.9.2, which is a formal statement of Theorem 4.2.2.

Theorem 4.9.5. *The following are equivalent:*

1. *There is no infinitely-often one-way function.*
2. *For every polynomials $b(n), s(n), t(n)$, and $t'(n)$, the class $\mathcal{F} = \{f\colon \{0,1\}^n \to \{0,1\}^{b(n)} : n \in \mathbb{N}\}$ is agnostically learnable in polynomial time on average under (unknown) $t'(n)$-time samplable distributions over $t(n)/s(n)$-time computable samplers with sample complexity $m(n,\epsilon,\delta) = O(s(n)\epsilon^{-2}\log\delta^{-1})$.*

Note that the sample complexity in item 2 is optimal when δ is constant, as observed in Section 4.9.4.

Proof. The implication item 2 $\Rightarrow$ item 1 is due to [GGM86; HILL99] and the observation in [Val84]. Thus, we only show the implication item 1 $\Rightarrow$ item 2.

Suppose that there is no infinitely-often one-way function (item 1). Then, there exists the universal extrapolation algorithm UE in Theorem 4.7.1.

First, we construct an agnostic learner as a cheating learner $L^{?}_{cheat}$ as follows: On input $1^{\epsilon^{-1}}, 1^b$ (where $\epsilon^{-1}, b \in \mathbb{N}$) and given access to distribution Label over $\{0,1\}^{\leq b}$, the learner $L^{?}_{cheat}$ obtains $q := (96)^2(b+1)\epsilon^{-2}\ln(192\epsilon^{-1})$ samples $y^1, \ldots, y^q$ from Label and outputs the most frequently

sampled label $\tilde{y} \in \{0,1\}^{\leq b}$, i.e., $\tilde{y} = y^{\tilde{i}}$ for $\tilde{i} = \arg\max_{i \in [q]} |\{j \in [q] : y^i = y^j\}|$. Trivially, $L^?_{cheat}$ halts in $\mathsf{poly}(\epsilon^{-1}, b)$ time.

We apply Theorem 4.9.2 for $L^?_{cheat}$ to obtain the learner L' that simulates $L^?_{cheat}$. We can show that L' satisfies the following property:

Claim 4.9.6. *There exists a polynomial $m_0(n, \epsilon^{-1}, \delta^{-1}) = O(s(n)\epsilon^{-2}\delta^{-2})$ such that for every $t'(n)$-time samplable distribution $\mathcal{D}$ over $t(n)/s(n)$-time computable samplers, every large enough $n \in \mathbb{N}$, every $\epsilon^{-1}, \delta^{-1} \in \mathbb{N}$,*

$$\Pr_{S,i,(x^1,y^1),\ldots,(x^{i-1},y^{i-1})}\left[\Pr_{(x,y)\sim S,L'}[L'(x^{<i}, x, 1^{\epsilon^{-1}}, 1^{b(n)}; 1^{\langle n,s(n),b(n),\tau,16\epsilon^{-1},\delta^{-1}\rangle}) \neq y] \leq \mathsf{opt}_{\mathcal{F}}(S) + \frac{\epsilon}{8}\right] \geq 1-\delta,$$

where $m = m_0(n, \epsilon^{-1}, \delta^{-1})$, $S \sim \mathcal{D}_n$, $i \sim [m]$, $(x^1, y^1), \ldots, (x^{i-1}, y^{i-1}) \sim S$, $x^{<i} = x^1 y^1 \cdots x^{i-1} y^{i-1}$, and $\tau = O(t'(n) + t(n)m)$.

First, we assume Claim 4.9.6 and show Theorem 4.9.5.

We construct an agnostic learner L for a fixed confidence error $\delta = 1/4$ with sample complexity $O(s(n)\epsilon^{-2})$. To reduce the confidence error $1/4$ to arbitrary $\delta \in (0, 1]$ given as a parameter, it suffices to repeat L $O(\log \delta^{-1})$ times with the accuracy error $\epsilon/2$ and outputs the best hypothesis by empirically estimating the accuracy error of each hypothesis within the approximation error $\pm\epsilon/2$ (see [HKLW88]). The time and sample complexity is affected only by the multiplicative factor $O(\log \delta^{-1})$ (note that the emprical estimation only requires additional $O(\epsilon^{-2} \log \delta^{-1})$ samples).

The construction of L is as follows: On input $1^n, 1^{\epsilon^{-1}}$ and given access to EX_S, where S is an unknown $t(n)/s(n)$-time computable sampler of example size n drawn from an unknown $t'(n)$-time samplable distribution $\mathcal{D}_n$, the learner L selects $i \sim [m]$, where $m = m_0(n, \epsilon^{-1}, 8)$ and m_0 is the polynomial in Claim 4.9.6, and obtains $i-1$ samples $(x^1, y^1), \ldots, (x^{i-1}, y^{i-1})$ from EX_S. Then, L selects a sufficiently long random string r and outputs a circuit (i.e., hypothesis) h_r that is taken $x \in \{0,1\}^n$ as input and outputs

$$y = L'(x^1 y^1 \cdots x^{i-1} y^{i-1}, x, 1^{\epsilon^{-1}}, 1^{b(n)}; 1^{\langle n,s(n),b(n),\tau,16\epsilon^{-1},8\rangle}; r),$$

where $\tau = O(t'(n) + t(n)m)$ indicated in Claim 4.9.6.

It is not hard to verify L halts in $\mathsf{poly}(n, \epsilon^{-1})$ time. In addition, the sample complexity is $m_0(n, \epsilon^{-1}, 8) = O(s(n)\epsilon^{-2})$. We also verify the correctness of L as follows. By Claim 4.9.6, with probability at least $7/8$ over the choice of $S, i, (x^1, y^1), \ldots, (x^{i-1}, y^{i-1})$, it holds that

$$0 \leq \Pr_{(x,y)\sim S,r}[h_r(x) \neq y] - \mathsf{opt}_{\mathcal{F}}(S) \leq \epsilon/8,$$

where the non-negativity follows from the definition of $\mathsf{opt}_{\mathcal{F}}(S)$. Thus, by Markov's inequality,

$$\Pr_r\left[\Pr_{(x,y)\sim S}[h_r(x) \neq y] - \mathsf{opt}_{\mathcal{F}}(S) \leq \epsilon\right] \geq 7/8.$$

By the union bound, with probability at least 1- 1/8 -1/8= 3/4 over the choice of S, randomness for L, and samples drawn from EX_S, the learner L outputs a hypothesis h_r that satisfies

$$\Pr_{(x,y)\sim S}[h_r(x) \neq y] \leq \mathsf{opt}_{\mathcal{F}}(S) + \epsilon.$$

In the remainder, we show Claim 4.9.6.

Proof of Claim 4.9.6. First, we analyze the performance of the cheating learner $L^{?}_{cheat}$. Let Label be an arbitrary distribution over $\{0,1\}^{\leq b}$. Let y^* be the label mostly generated according to Label, i.e., $y^* := \arg\max_{y\in\{0,1\}^{\leq b}} \mathsf{Label}(y)$ (breaking ties arbitrarily). Remember that $L^{\mathsf{Label}}_{cheat}$ collects $q := (96)^2(b+1)\epsilon^{-2}\ln(192\epsilon^{-1})$ samples $y^1,\dots,y^q$. By Hoeffding's inequality, for each $y\in\{0,1\}^{\leq b}$, the estimated outcome probability $\tilde{p}_y = |\{i\in[q] : y^i = y\}|/q$ satisfies that $\mathsf{Label}(y) - \epsilon/96 \leq \tilde{p}_y \leq \mathsf{Label}(y) + \epsilon/96$ with probability at least $1 - 2e^{2q(\epsilon/96)^2} \geq 1 - (\epsilon/96)\cdot 2^{-(b+1)}$. By the union bound, $\mathsf{Label}(y) - \epsilon/96 \leq \tilde{p}_y \leq \mathsf{Label}(y) + \epsilon/96$ holds for all $y\in\{0,1\}^{\leq b}$ with probability at least $1-\epsilon/96$. Under this event, the probability that the output $\tilde{y}$ of $L^{\mathsf{Label}}_{cheat}$ corresponds to $y\sim\mathsf{Label}$ is at least

$$\mathsf{Label}(\tilde{y}) \geq \tilde{p}_{\tilde{y}} - \epsilon/96 \geq \tilde{p}_{y^*} - \epsilon/96 \geq \mathsf{Label}(y^*) - \epsilon/48.$$

In this case, we have that

$$\begin{aligned}\Pr_{\mathsf{Label},y\sim\mathsf{Label}}[L^{\mathsf{Label}}_{cheat}(1^\epsilon,1^b)\neq y | \forall \tilde{p}_y \in \mathsf{Label}(y)\pm\epsilon/96] &\leq \Pr_{y\sim\mathsf{Label}}[y\neq y^*] + \epsilon/48\\ &= \min_{\tilde{y}\in\{0,1\}^{\leq b}} \Pr_{y\sim\mathsf{Label}}[y\neq\tilde{y}] + \epsilon/48.\end{aligned}$$

Therefore, for every $\epsilon^{-1}, b\in\mathbb{N}$ and every Label over $\{0,1\}^{\leq b}$,

$$\Pr_{\mathsf{Label},y\sim\mathsf{Label}}[L^{\mathsf{Label}}_{cheat}(1^\epsilon,1^b)\neq y] = \min_{\tilde{y}\in\{0,1\}^{\leq b}} \Pr_{y\sim\mathsf{Label}}[y\neq\tilde{y}] + \epsilon/48 + \epsilon/96 \leq \min_{\tilde{y}\in\{0,1\}^{\leq b}} \Pr_{y\sim\mathsf{Label}}[y\neq\tilde{y}] + \epsilon/32.$$

Thus, for every $\epsilon^{-1}, b\in\mathbb{N}$,

$$\Delta_{L_{cheat}}(1^\epsilon,1^b,b) := \sup_{\mathsf{Label}}\left(\Pr_{\mathsf{Label},y\sim\mathsf{Label}}[L^{\mathsf{Label}}_{cheat}(1^\epsilon,1^b)\neq y] - \min_{\tilde{y}\in\{0,1\}^{\leq b}} \Pr_{y\sim\mathsf{Label}}[y\neq\tilde{y}]\right) \leq \epsilon/32.$$

Now, we analyze the performance of L'. Let $\mathcal{D}$ be an arbitrary $t'(n)$-time samplable distribution over $t(n)/s(n)$-time computable samplers. Let m_0 be the polynomial in Theorem 4.9.2.

We define a distribution family $\mathcal{E} = \{\mathcal{E}_{n,z}\}_{n\in\mathbb{N},z\in\{0,1\}^*}$, where $\mathcal{E}_{n,z}$ is a distribution of an *infinitely* long string $x^1x^2x^3\cdots$, where for each $i\in\mathbb{N}$, $x^i \sim U^{t(n)}(s,r^i)$ for a uniformly random seed $r^i\sim\{0,1\}^t$ (note that if s is a description of a sampler, then each x^i corresponds to a sample). Then, for every $n,\epsilon^{-1},\delta^{-1}\in\mathbb{N}$ and $m_0 := m_0(\log n, s(n), 1, 16\epsilon^{-1},\delta^{-1})$, the prefix $x^1\cdots x^{m_0}$ of $\mathcal{D}_{n,s_0}$ is samplable in $O(m_0\cdot t(n))$ time. Therefore, by Theorem 4.9.2 (for $\mathcal{D}$ and $\mathcal{E}$), for every large enough $n\in\mathbb{N}$ and every $\epsilon^{-1},\delta^{-1}$ (note that we choose $a=n$ and $b=b(n)$), and for large enough $\tau = \max\{t'(n), O(m_0\cdot t(n))\}$,

$$\Pr_{S,i,\{(x^j,y^j)\}_{j=1}^{i-1}}\left[\Pr_{(x,y)\sim S,L'}\left[L'(x^{<i},x,1^\epsilon,1^{b(n)};1^N)\neq y\right] \leq \mathop{\mathbb{E}}_{x\sim S^{(1)}}\left[\min_{y^*}\Pr_{y\sim S^{(2)}_{|x}}[y^*=y]\right] + 2\Delta_{L_{cheat}} + \frac{\epsilon}{16}\right] \geq 1-\delta,$$

where $S\sim\mathcal{D}_n$, $i\sim[m_0]$, and $(x^1,y^1),\dots,(x^{i-1},y^{i-1})\sim S$, $N = \langle n, s(n), b(n), \tau, 16\epsilon^{-1},\delta^{-1}\rangle$, $x^{<i} = x^1y^1\cdots x^{i-1}y^{i-1}$, $S^{(2)}_{|x}$ is a conditional distribution of $S^{(2)}$ given $x\sim S^{(1)}$, and $\Delta_{L_{cheat}} = \Delta_{L_{cheat}}(1^\epsilon,1^b,b) \leq \epsilon/32$. It is easy to verify that

$$\mathop{\mathbb{E}}_{x\sim S^{(1)}}\left[\min_{y^*\in\{0,1\}^{<b(n)}}\Pr_{y\sim S^{(2)}_{|x}}[y^*=y]\right] = \mathsf{opt}_{\mathcal{F}}(S).$$

Thus, we conclude that

$$\Pr_{S,i,\{(x^j,y^j)\}_{j=1}^{i-1}}\left[\Pr_{(x,y)\sim S,L'}\left[L'(x^{<i},x,1^\epsilon,1^{b(n)};1^N)\neq y\right]\leq \mathsf{opt}_{\mathcal{F}}(S)+\epsilon/8\right]\geq 1-\delta.$$

The sample complexity m_0 is bounded above by

$$m_0(\log n, s(n), 1, 16\epsilon^{-1}, \delta^{-1}) = O((\log n + s(n))\epsilon^{-2}\delta^{-2}) = O(s(n)\epsilon^{-2}\delta^{-2}),$$

where we assume that $\log n \leq s(n)$; otherwise, a learner can try all possible $t(n)/s(n)$-time computable samplers in $t(n)\cdot 2^{O(s(n))}\leq \mathsf{poly}(n)$ time and output the best hypothesis by the standard empirical estimation of the accuracy error. □

□

4.9.3 Universal Agnostic Learning for General Loss

In this section, we consider agnostic learning for general polynomial-time computable loss functions.

Theorem 4.9.7. *The following are equivalent:*

1. *There is no infinitely-often one-way function.*
2. *For every $b(n)=O(\log n)$, every polynomials $s(n),t(n),t'(n)$, and $c(n)$, and every polynomial-time computable loss function l with $l(\tilde{y},y)\leq c(n)$ for each $n\in\mathbb{N}$ and each $\tilde{y},y\in\{0,1\}^{\leq b(n)}$, the class $\mathcal{F}=\{f\colon\{0,1\}^n\to\{0,1\}^{b(n)}:n\in\mathbb{N}\}$ is agnostically learnable for the loss function l in polynomial time on average under (unknown) $t'(n)$-time samplable distributions over $t(n)/s(n)$-time computable samplers with sample complexity $m(n,\epsilon,\delta)=O(s(n)c(n)^2\epsilon^{-2}\log\delta^{-1})$.*

Proof. The implication item 2 ⇒ item 1 is due to [GGM86; HILL99] and the observation in [Val84]. Thus, we only show the implication item 1 ⇒ item 2.

The outline of the proof is the same as Theorem 4.9.5. The only difference is the construction of the cheating learner $L^?_{cheat}$.

Let $l\colon\{0,1\}^*\times\{0,1\}^*\to\mathbb{R}_{\geq 0}$ be an arbitrary polynomial-time computable loss function such that $l(\tilde{y},y)\leq c(n)$ for each $n\in\mathbb{N}$ and each $\tilde{y},y\in\{0,1\}^{\leq b(n)}$, where $b(n)=O(\log n)$ and $c(n)=\mathsf{poly}(n)$. We construct a cheating learner $L^?_{cheat}$ for minimizing expected loss with respect to l with additive error $\epsilon/32$ for all distributions Label over $\{0,1\}^{\leq b(n)}$.

On input $1^n,1^{\epsilon^{-1}}$ and given access to a distribution Label over $\{0,1\}^{\leq b(n)}$, the learner L_{cheat} obtains $q:=(64c(n)2^{b(n)+1})^2(b(n)+1)\epsilon^{-2}\ln(128\epsilon^{-1}c(n))$ samples $y^1,\dots y^q$ from Label and computes $\tilde{p}_y=|\{j\in[q]:y^j=y\}|/q$ for each $y\in\{0,1\}^{\leq b(n)}$ (notice that it takes only $O(q2^{b(n)})=\mathsf{poly}(n,\epsilon^{-1})$ times). Note that these estimated probabilities determine an estimated distribution $\tilde{Y}$ over $\{0,1\}^{\leq b(n)}$, where each y is drawn from $\tilde{Y}$ with probability $\tilde{p}_y$. The leaner L_{cheat} minimizes the expected loss under $\tilde{Y}$, i.e., L_{cheat} computes $\tilde{l}_\alpha=\mathbb{E}_{y\sim\tilde{Y}}[l(\alpha,y)]$ for each $\alpha\in\{0,1\}^{\leq b(n)}$ and outputs $\tilde{\alpha}=\arg\min_{\alpha\in\{0,1\}^{\leq b(n)}}\tilde{l}_\alpha$ (notice that it takes only $O(2^{b(n)})=\mathsf{poly}(n)$ times).

We show that for every $n,\epsilon^{-1}\in\mathbb{N}$ and every distribution Label over $\{0,1\}^{\leq b(n)}$,

$$\mathop{\mathbb{E}}_{\mathsf{Label},y\sim\mathsf{Label}}[l(L^{\mathsf{Label}}_{cheat}(1^n,1^{\epsilon^{-1}}),y)]\leq \min_{\alpha\in\{0,1\}^{\leq b}}\mathop{\mathbb{E}}_{y\sim\mathsf{Label}}[l(\alpha,y)]+\epsilon/32. \tag{4.9}$$

This implies Theorem 4.9.7 by the same argument as Theorem 4.9.5, i.e., we apply Theorem 4.9.2 for L_{cheat} to obtain the learner L' that simulates L_{cheat}, and then we construct an agnostic learner L (with fixed confidence error $\delta = 1/4$) that selects $i \sim [m]$ and randomness r, where $m = O(s(n)c(n)^2\epsilon^{-2})$ indicated in Theorem 4.9.2, collects $i-1$ samples $(x^1, y^1), \ldots, (x^{i-1}, y^{i-1})$ from EX_S, and outputs h_r that takes $x \in \{0,1\}^n$ as input and outputs

$$y = L'(x^1 y^1 \cdots x^{i-1} y^{i-1}, x, 1^n, 1^{\epsilon^{-1}}; 1^{\langle n, s(n), b(n), \tau, 32\epsilon^{-1}, 8\rangle}; r),$$

for $\tau = O(t'(n) + t(n)m)$. The correctness of L holds in the same way as Theorem 4.9.5 if inequality (4.9) holds.

We verify inequality (4.9). Let $\alpha^* = \arg\min_{\alpha \in \{0,1\}^{\le b(n)}} \mathbb{E}_{y \sim \mathsf{Label}}[l(\alpha, y)]$. Remember that $L_{cheat}^{\mathsf{Label}}$ collects $q := (64c(n)2^{b(n)+1})^2(b(n)+1)\epsilon^{-2}\ln(128\epsilon^{-1}c(n))$ samples $y^1, \ldots, y^q$. By Hoeffding's inequality, for each $y \in \{0,1\}^{\le b(n)}$, the estimated outcome probability $\tilde{p}_y = |\{i \in [q] : y^i = y\}|/q$ satisfies that $\mathsf{Label}(y) - \epsilon/(64c(n)2^{b(n)+1}) \le \tilde{p}_y \le \mathsf{Label}(y) + \epsilon/(64c(n)2^{b(n)+1})$ with probability at least $1 - 2e^{2q(\epsilon/64c(n)2^{b(n)+1})^2} \ge 1 - (\epsilon/64c(n)) \cdot 2^{-(b(n)+1)}$. By the union bound, $\mathsf{Label}(y) - \epsilon/(64c(n)2^{b(n)+1}) \le \tilde{p}_y \le \mathsf{Label}(y) + \epsilon/(64c(n)2^{b(n)+1})$ holds for all $y \in \{0,1\}^{\le b(n)}$ with probability at least $1 - \epsilon/64c(n)$. Under this event, the output $\tilde{\alpha} \in \{0,1\}^{\le b(n)}$ of $L_{cheat}^{\mathsf{Label}}$ satisfies that

$$\begin{aligned}
\mathop{\mathbb{E}}_{y\sim\tilde{Y}}[l(\tilde{\alpha}, y)] \le \mathop{\mathbb{E}}_{y\sim\tilde{Y}}[l(\alpha^*, y)] &= \sum_{y\in\{0,1\}^{\le b(n)}} \tilde{p}_y \cdot l(\alpha^*, y) \\
&\le \sum_{y\in\{0,1\}^{\le b(n)}} (\mathsf{Label}(y) + \frac{\epsilon}{64c(n)2^{b(n)+1}}) \cdot l(\alpha^*, y) \\
&\le \sum_{y\in\{0,1\}^{\le b(n)}} \mathsf{Label}(y) l(\alpha^*, y) + \frac{\epsilon \cdot |\{0,1\}^{\le b(n)}|}{64c(n)2^{b(n)+1}} \cdot \max_{y\in\{0,1\}^{\le b(n)}} l(\alpha^*, y) \\
&\le \mathop{\mathbb{E}}_{y\sim\mathsf{Label}}[l(\alpha^*, y)] + \frac{\epsilon}{64} \\
&= \min_{\alpha\in\{0,1\}^{\le b(n)}} \mathop{\mathbb{E}}_{y\sim\mathsf{Label}}[l(\alpha, y)] + \frac{\epsilon}{64}.
\end{aligned}$$

Let B be the event that there exists $y \in \{0,1\}^{\le b(n)}$ such that $\mathsf{Label}(y) - \epsilon/(64c(n)2^{b(n)+1}) \le \tilde{p}_y \le \mathsf{Label}(y) + \epsilon/(64c(n)2^{b(n)+1})$ does not hold. Then, we have

$$\begin{aligned}
\mathop{\mathbb{E}}_{\mathsf{Label}, y\sim\mathsf{Label}}[l(L_{cheat}^{\mathsf{Label}}(1^n, 1^{\epsilon^{-1}}), y)] &\le \Pr[\neg B] \cdot \left(\min_{\alpha\in\{0,1\}^{\le b(n)}} \mathop{\mathbb{E}}_{y\sim\mathsf{Label}}[l(\alpha, y)] + \frac{\epsilon}{64} \right) + \Pr[B] \cdot \max_{\alpha, y\in\{0,1\}^{\le b(n)}} l(\alpha, y) \\
&\le 1 \cdot \left(\min_{\alpha\in\{0,1\}^{\le b(n)}} \mathop{\mathbb{E}}_{y\sim\mathsf{Label}}[l(\alpha, y)] + \frac{\epsilon}{64} \right) + \frac{\epsilon}{64c(n)} \cdot c(n) \\
&= \min_{\alpha\in\{0,1\}^{\le b(n)}} \mathop{\mathbb{E}}_{y\sim\mathsf{Label}}[l(\alpha, y)] + \frac{\epsilon}{32}.
\end{aligned}$$

□

4.9.4 Lower Bound on Sample Complexity in Agnostic Learning on Average

In this section, we show that the dependence of s and ϵ^{-1} on the sample complexity of our agnostic learner (in Theorems 4.2.2 and 4.9.5) is optimal, particularly in learning parities on average with

noise over unit vectors. The proof is essentially same as the proof of the fundamental theorem of statistical learning (cf. [SB14, Sections 6 and 28]) except that we consider a natural average-case analogue of shattered sets.

We review the problem of learning parities with noise (LPN). For every $\alpha \in \{0,1\}^n$, let $\chi_\alpha \colon \{0,1\}^n \to \{0,1\}$ denote a parity function defined as $\chi_\alpha(x) = \langle x, \alpha \rangle_{\mathbb{F}_2}$, where $\langle , \rangle_{\mathbb{F}_2}$ represents the inner product on $\mathbb{F}_2$. For each $n \in \mathbb{N}$, $\alpha \in \{0,1\}^n$, $\gamma \in [0, 1/2]$, and $S \subseteq \{0,1\}^n$, we define a distribution $\mathsf{LPN}^S_{\alpha,\gamma}$ over samples in $\{0,1\}^n \times \{0,1\}$ as the distribution of $(x, \chi_\alpha(x) \oplus \xi)$ for $x \sim S$ and $\xi \sim \mathsf{Ber}(1/2 - \gamma)$, where $\mathsf{Ber}(1/2 - \gamma)$ represents the Bernoulli distribution with parameter $1/2 - \gamma$, i.e., $\xi = 1$ (resp. $\xi = 0$) with probability $1/2 - \gamma$ (resp. $1/2 + \gamma$). It is easily verified that, for every $\gamma \in [0, 1/2]$ and every $S \subseteq \{0,1\}^n$,

$$\mathsf{opt}_{\mathcal{F}}(\mathsf{LPN}^S_{\alpha,\gamma}) = \min_{f \colon \{0,1\}^n \to \{0,1\}} \Pr_{(x,b) \sim \mathsf{LPN}^S_{\alpha,\gamma}} [f(x) \neq b] = \frac{1}{2} - \gamma.$$

We mainly focus on the specific subset $S^e_n = \{e^1, \ldots, e^{\lfloor n/2 \rfloor}\} \subseteq \{0,1\}^n$, where each $e^j \in \{0,1\}^n$ is the j-th unit vector, i.e., $e^j_i = 1$ iff $i = j$. Note that for every $c \in \mathbb{N}$, the distribution $\mathsf{LPN}^{S^e_{|\alpha|}}_{\alpha,2^{-c}}$ is samplable in time $O(|\alpha| \cdot c)$ when α is given as advice.

We show the following matching lower bound on sample complexity for LPN over S^e_n.

Theorem 4.9.8. *Suppose that a (possibly not efficient) agnostic learner L satisfies that for every large enough $n \in \mathbb{N}$, every $\epsilon^{-1} \in \mathbb{N}$, and every $c \in \mathbb{N}$ with $c \leq \log^2 n$,*

$$\Pr_{\alpha \sim \{0,1\}^n, \mathsf{LPN}^{S^e_n}_{\alpha,2^{-c}}} \left[L^{\mathsf{LPN}^{S^e_n}_{\alpha,2^{-c}}}(1^n, 1^{\epsilon^{-1}}) \text{ outputs } h \text{ s.t. } \Pr_{(x,b) \sim \mathsf{LPN}^{S^e_n}_{\alpha,2^{-c}}} [h(x) \neq b] \leq \left(\frac{1}{2} - 2^{-c}\right) + \epsilon \right] \geq \frac{7}{8}.$$

Then, the sample complexity m_L of L must satisfy that $m_L(n, \epsilon) = \Omega(n\epsilon^{-2})$ (note that the secret information is represented by $s := |\alpha| = n$ bits).

Theorem 4.9.8 is obtained in the same way as the fundamental theorem of statistical learning, where we use the simple observation that parity functions shatter S^e_n *on average* over the choice of parity functions (instead of the argument of the VC dimension). Here, we only present the proof outline. For the detailed argument, we refer the reader to the proof in [SB14, Section 28.2.2].

Proof sketch. Suppose that the sample complexity $m_L(n, \epsilon)$ satisfies that $m_L(n, \epsilon) < 8\lfloor n/2 \rfloor \epsilon^{-2}$. Below, we derive a contradiction to show $m_L(n, \epsilon) \geq 8\lfloor n/2 \rfloor \epsilon^{-2} = \Omega(n\epsilon^{-2})$. For readability, we omit the superscript S^e_n from $\mathsf{LPN}^{S^e_n}_{\alpha,\rho/2}$ in this proof.

Fix large enough $n \in \mathbb{N}$ and $k \in \mathbb{N}$ with $k \in (\log 8\sqrt{2}, \log^2 n + 2)$ arbitrarily. Let $m = \lfloor n/2 \rfloor$. Let $\epsilon = 2^{-k}$ and $\rho = 8\epsilon$ (note that $\rho < 1/\sqrt{2}$). Let $c = -\log \rho = k - 3$. Since $c + 1 \leq \log^2 n$, the learner L satisfies that

$$\Pr_{\alpha \sim \{0,1\}^n, \mathsf{LPN}_{\alpha,\rho/2}} \left[L^{\mathsf{LPN}_{\alpha,\rho/2}}(1^n, 1^{\epsilon^{-1}}) \text{ outputs } h \text{ s.t. } \Pr_{(x,b) \sim \mathsf{LPN}_{\alpha,\rho/2}} [h(x) \neq b] \leq (1 + \rho)/2 + \epsilon \right] \geq \frac{7}{8},$$

and $L(1^n, 1^{\epsilon^{-1}})$ makes only at most $8m\epsilon^{-2}$ queries. Without loss of generality, we assume that $L(1^n, 1^{\epsilon^{-1}})$ makes *exactly* $M := 8m\epsilon^{-2}$ queries, and M samples $S_\alpha := \{(x^i, \chi_\alpha(x^i) \oplus \xi^i)\}_{i=1}^M$ are given as auxiliary input, where $\alpha \sim \{0,1\}^n$, $x^i \sim S^e_n$, and $\xi^i \sim \mathsf{Ber}((1 + \rho)/2)$ for each i.

For each $\alpha \in \{0,1\}^n$, let $\mathsf{opt}_\alpha := \mathsf{opt}_{\mathcal{F}}(\mathsf{LPN}_{\alpha,\rho/2}) = (1-\rho)/2$. For each hypothesis $h\colon \{0,1\}^n \to \{0,1\}$, let $\ell_\alpha(h)$ be the error probability of h, i.e.,

$$\ell_\alpha(h) := \Pr_{(x,b)\sim\mathsf{LPN}_{\alpha,\rho/2}}[h(x) \neq b] = \frac{1+\rho}{2} \cdot \frac{|\{i \in [m] : h(e^i) \neq \alpha_i\}|}{m} + \frac{1-\rho}{2} \cdot \frac{|\{i \in [m] : h(e^i) = \alpha_i\}|}{m}.$$

Therefore,

$$\ell_\alpha(h) - \mathsf{opt}_\alpha = \rho \cdot \frac{|\{i \in [m] : h(e^i) \neq \alpha_i\}|}{m}. \tag{4.10}$$

Let $L(S_\alpha)$ denote the hypothesis produced by $L(1^n, 1^{\epsilon^{-1}})$ given a sample set S_α. Then, we have

$$\Pr_{\alpha,S_\alpha}[\ell_\alpha(L(S_\alpha)) - \mathsf{opt}_\alpha > \epsilon] \leq 1/8.$$

Now, we consider another learning algorithm L^* that is given S_α and produces a hypothesis $L^*(S_\alpha)\colon S_n^e \to \{0,1\}$ such that, for each $e^i \in S_n^e$, the value of $L^*(S_\alpha)(e^i)$ is the majority among the labels of e^i in the sample set S_α (breaking ties arbitrarily). Then, L^* is the optimal learner (cf. [SB14, Lemma 28.1]), i.e.,

$$\Pr_{\alpha,S_\alpha}[\ell_\alpha(L^*(S_\alpha)) - \mathsf{opt}_\alpha > \epsilon] \leq \Pr_{\alpha,S_\alpha}[\ell_\alpha(L(S_\alpha)) - \mathsf{opt}_\alpha > \epsilon] \leq 1/8. \tag{4.11}$$

Furthermore,

$$\begin{aligned}\mathop{\mathbb{E}}_{\alpha,S_\alpha}[\ell_\alpha(L^*(S_\alpha)) - \mathsf{opt}_\alpha] &= \frac{\rho}{m} \mathop{\mathbb{E}}_{\alpha,S_\alpha}\left[|\{i \in [m] : L^*(S_\alpha)(e^i) \neq \alpha_i\}|\right] \\ &= \frac{\rho}{m} \sum_{i=1}^{m} \Pr_{\alpha,S_\alpha}[L^*(S_\alpha)(e^i) \neq \alpha_i].\end{aligned}$$

By Slud's inequality and careful calculations (see [SB14, Section 28.2.2]), the right-hand side is bounded below as

$$\frac{\rho}{m} \sum_{i=1}^{m} \Pr_{\alpha,S_\alpha}[L^*(S_\alpha)(e^i) \neq \alpha_i] \geq \frac{\rho}{2}\left(1 - \sqrt{2\rho^2 M/m}\right),$$

where we use the fact that $\rho < 1/\sqrt{2}$.

Since $M < 8m\epsilon^{-2} = m/(8\rho^2)$,

$$\mathop{\mathbb{E}}_{\alpha,S_\alpha}[\ell_\alpha(L^*(S_\alpha)) - \mathsf{opt}_\alpha] \geq \frac{\rho}{2}\left(1 - \sqrt{2\rho^2 M/m}\right) > \frac{\rho}{2}\left(1 - \frac{\rho}{2}\right) = \frac{\rho}{2} - \frac{\rho^2}{4} \geq \frac{\rho}{4} = 2\epsilon.$$

By equation (4.10), it holds that $\ell_\alpha(L^*(S_\alpha)) - \mathsf{opt}_\alpha \leq \rho$ for every α and S_α. Therefore, we have

$$\Pr_{\alpha,S_\alpha}[\ell_\alpha(L^*(S_\alpha)) - \mathsf{opt}_\alpha > \epsilon] > \frac{1}{8}, \tag{4.12}$$

otherwise,

$$\mathop{\mathbb{E}}_{\alpha,S_\alpha}[\ell_\alpha(L^*(S_\alpha)) - \mathsf{opt}_\alpha] \leq \epsilon \cdot 1 + \frac{1}{8} \cdot \rho = 2\epsilon.$$

Inequality (4.12) contradicts inequality (4.11). □

Chapter 5

Learning in Pessiland II: More Dichotomies between Learning and Cryptography

This chapter gives further dichotomy between learning and cryptography, where our focus lies mainly on stronger learning-theoretic consequences in a restricted setting of average-case learning and with an additional assumption. In Section 5.1, we show the equivalence between the existence of OWF and the average-case hardness of MINLT in the BFKL model [BFKL93] under the additional derandomization assumption. Remember that the problem MINLT is known to be NP-complete (see Section 2.6). In Section 5.2, we study relationships between hardness of learning and an *auxiliary-input* one-way function.

5.1 MINLT vs. One-Way Functions

In this section, we show that the search version of MINLT is efficiently solvable on average in the BFKL model [BFKL93] under the non-existence of OWF and the standard derandomization assumption.

First, we review the problem MINLT. We introduce additional notions. In this section, we may call a distribution over $\{0,1\}^n \times \{0,1\}$ a *sampler*. For any $m \in \mathbb{N}$ and any distribution family $\mathcal{D} = \{\mathcal{D}_n\}_{n\in\mathbb{N}}$, where each $\mathcal{D}_n$ is a distribution on samplers over $\{0,1\}^n \times \{0,1\}$, we let $\mathcal{D}^m = \{\mathcal{D}_n^m\}_{n\in\mathbb{N}}$ denote a distribution family over sample sets such that each $\mathcal{D}_n^m$ is the distribution of a sample set $\{(x^1,b^1),\dots,(x^m,b^m)\}$, where $(x^i,b^i) \sim S$ for each $i \in [m]$ and $S \sim \mathcal{D}_n$ (note that the sampler S is selected only once). We also use the notation $(\boldsymbol{x},\boldsymbol{b}) \sim \mathcal{D}_n^m$ to represent that $\boldsymbol{x} = (x^1,\dots,x^m)$ and $\boldsymbol{b} = (b^1,\dots,b^m)$ for $\{(x^1,b^1),\dots,(x^m,b^m)\} \sim \mathcal{D}_n^m$.

We define LT-complexity of sample sets.

Definition 5.1.1 (LT-complexity [Ko91])**.** *Let $X = \{(x^1,b^1),\dots,(x^m,b^m)\}$ be a sample set, where $x^i \in \{0,1\}^*$ and $b^i \in \{0,1\}$ for each $i \in [m]$. For every $t \in \mathbb{N}$, the t-time-bounded LT-complexity of X is denoted by $\mathrm{LT}^t(X)$ and defined as*

$$\mathrm{LT}^t(X) := \min\{|\Pi| : \Pi \in \{0,1\}^* \textit{such that } U^t(\Pi,x^i) = b^i \textit{ for all } i \in [m]\}.$$

Now, we define MINLT as a problem that asks LT-complexity of a given sample set.

Definition 5.1.2 (MINLT [Ko91]). *For every $t \in \mathbb{N}$, we define the language* MINLT$[t]$ *as follows:*

$$\mathrm{MINLT}[t] = \left\{(X, 1^s) : n, m \in \mathbb{N}, X \in (\{0,1\}^n \times \{0,1\})^m, \mathrm{LT}^t(X) \leq s\right\}.$$

We define the search version of MINLT$[t]$ *as a problem that asks, for a given $X = \{(x^1, b^1), \ldots, (x^m, b^m)\}$, where $x^i \in \{0,1\}^n$ and $b^i \in \{0,1\}$ for each $i \in [m]$, to find $\Pi \in \{0,1\}^*$ such that $|\Pi| = \mathrm{LT}^t(X)$ and $U^t(\Pi, x^i) = b^i$ holds for all $i \in [m]$.*

In this section, we show the following equivalence result.

Theorem 5.1.3. *Under the standard derandomization assumption (Hypothesis 5.1.6), the following are equivalent:*

1. *There exists no infinitely-often one-way function;*
2. *For every polynomial-time evaluatable concept class $\mathscr{C}$, there exist a polynomial-time randomized algorithm L and a polynomial τ_0 such that for every polynomial $t(n)$, every unknown $t(n)$-time samplable distribution $\mathcal{E}$ over examples, every unknown $t(n)$-time samplable distribution $\mathcal{F}$ over $\mathscr{C}$, every sufficiently large $n \in \mathbb{N}$, and every $m, \delta^{-1}, \tau \in \mathbb{N}$ with $\tau \geq \tau_0(n, m, t(n))$, the algorithm $L(\text{-}; 1^{\langle n, m, \delta^{-1}, t(n), \tau\rangle})$ solves the search version of* MINLT$[\tau]$ *on average in the BFKL model with respect to $\mathcal{E}$ and $\mathcal{F}$ with error probability at most δ.*

Before presenting the proof, we mention related works. Remember that MINLT is recently shown to be NP-complete by Hirahara [Hir22a]. In meta-complexity, there are several works that base OWF on the average-case hardness of NP-complete problem on the *uniform* distribution over instances. Liu and Pass [LP22] presented the characterization by the average-case hardness of MINcKT on the *uniform* distribution, where MINcKT is the problem of asking $\mathrm{K}^t(x|y)$ for given strings x and y (technically, the time-bound t must be smaller enough than $|y|$ in their result). Allender, Cheraghchi, Myrisiotis, Tirumala, and Volkovich [ACMTV21] presented the characterization by the average-case hardness of McKTP on the *uniform* distribution, where McKTP is the problem of asking another related meta-complexity notion called the conditional KT-complexity. Compared with the previous work, we consider the average-case hardness on more general distributions in the BFKL model than the uniform distribution (instead, we require an additional derandomization assumption). Note that the BFKL model was the most standard and general average-case formulation of learning before this thesis.

In this section, we only present the proof of Theorem 5.1.3 (1 $\Rightarrow$ 2), and the converse follows from the well-known results [Val84; GGM84; HILL99].

First, we present a meta-theorem. Roughly speaking, the meta-theorem shows that if there exists no infinitely-often one-way function, then we can construct an algorithm that solves MINLT$[\tau]$ on average under any samplable distribution on samplers that satisfies a time-bounded LT-complexity analogue of the coding theorem.

Lemma 5.1.4. *If there exists no infinitely-often one-way function, then for every constant $C > 0$, there exists a randomized algorithm L such that for every polynomials $t(n), \sigma(n)$ and every (unknown) $t(n)$-samplable distribution $\mathcal{D} = \{\mathcal{D}_n\}_{n\in\mathbb{N}}$ on samplers, for every sufficiently large $n \in \mathbb{N}$, for every $m, \delta^{-1}, \tau \in \mathbb{N}$, the algorithm $L(\text{-}; 1^{\langle n, m, \delta^{-1}, t(n), \sigma(n), \tau\rangle})$ solves the search version of* MINLT$[\tau]$ *on average under $\mathcal{D}^m$ with error probability at most δ as long as $\mathcal{D}$ satisfies that, for all $\gamma^{-1} \in \mathbb{N}$,*

$$\Pr_{X=(\boldsymbol{x},\boldsymbol{b})\sim\mathcal{D}_n^m}\left[\mathrm{LT}^\tau(X) \leq \min\{-\log \mathcal{D}_n^m(\boldsymbol{b}|\boldsymbol{x}) + C(\log\tau + \log\gamma^{-1}), \sigma(n)\}\right] \geq 1 - \gamma,$$

where $\mathcal{D}_n^m(\boldsymbol{b}|\boldsymbol{x}) = \Pr_{(\boldsymbol{x}',\boldsymbol{b}')\sim\mathcal{D}_n^m}[\boldsymbol{b}' = \boldsymbol{b}|\boldsymbol{x}' = \boldsymbol{x}]$.

Proof. Let $t_U(n)$ be a simulation overhead function of the universal Turing machine U, i.e., for every $t \in \mathbb{N}$, every Turing machine M, and every input $x \in \{0,1\}^*$, if $M(x)$ halts in t time, then $U^{t_U(t)}(M,x) = M(x)$.

We define a polynomial-time-computable family $f = \{f_n\colon \{0,1\}^{\mathsf{poly}(n)} \to \{0,1\}^{\mathsf{poly}(n)}\}$ as

$$f_{\langle n,m,t,\sigma,\tau\rangle}(i,j,\Pi_f,\Pi_e,s^0,s^1,\dots,s^m) = (i,x^1,U^\tau((\Pi_f)_{[i]},x^1),\dots,x^m,U^\tau((\Pi_f)_{[i]},x^m)),$$

where $i \in [\sigma]$, $j \in [n]$, $\Pi_f,\Pi_e \in \{0,1\}^n$, $s^0,s^1,\dots,s^m \in \{0,1\}^{t_U(t)}$, and $x^1,\dots x^m$ are determined as follows: for each $k \in [m]$,

$$S = U^{t_U(t)}((\Pi_e)_{[j]},\langle 1^n,s^0\rangle)$$
$$x^k = \begin{cases} \mathsf{CircEval}(S,s^k) & \text{if } S \text{ is a description of a circuit of output length } n \\ 0^n & \text{otherwise,} \end{cases}$$

where $\mathsf{CircEval}$ is the standard circuit evaluation algorithm that runs in polynomial time. Without loss of generality, we assume that the circuit size of S above is at most $t_U(t)$ because its description is printed in $t_U(t)$ time.

Since we execute U in polynomial time in above, f is a polynomial-time-computable family. We assume that the input to f_n is given as a concatenated string. For each $n,m,t,\sigma,\tau \in \mathbb{N}$, let $r(n,m,t,\sigma,\tau)$ be the input size of $f_{\langle n,m,t,\sigma,\tau\rangle}$, i.e., $r(n,m,t,\sigma,\tau) = \lceil\log\sigma\rceil + \lceil\log n\rceil + 2n + t_U(t)(m+1)$.

By Proposition 4.6.6 and the assumption that there exists no infinitely-often one-way function, there exists a randomized polynomial-time algorithm A such that for every $n,m,t,\sigma,\tau \in \mathbb{N}$ and every $\delta^{-1} \in \mathbb{N}$,

$$\Pr\left[A(f_{\langle n,m,t,\sigma,\tau\rangle}(U_{r(n,m,t,\sigma,\tau)});1^{\langle n,m,t,\sigma,\tau\rangle},1^{\delta^{-1}}) \notin f^{-1}_{\langle n,m,t,\sigma,\tau\rangle}(f_{\langle n,m,t,\sigma,\tau\rangle}(U_{r(n,m,t,\sigma,\tau)}))\right] \le \delta.$$

We construct a randomized algorithm L in the theorem from A. For given parameters $n,m,\delta^{-1},t,\sigma,\tau \in \mathbb{N}$ and a sample set $X = \{(x^i,b^i)\}_{i=1}^m$, the algorithm L executes

$$A(i,x^1,b^1,\dots,x^m,b^m;1^{\langle n,m,t,\sigma,\tau\rangle},1^{\delta'^{-1}})$$

for each $i \in [\sigma]$ and for δ' defined as

$$\delta' = \frac{1}{2}\cdot\frac{(\delta/2)^C}{n^2\sigma\tau^C}\frac{\delta}{2},$$

where $C > 0$ is the constant in the theorem. For each i, if A returns some inverse element $X^i = (i,j^i,\Pi_f^i,\Pi_e^i,s^{i,0},s^{i,1},\dots,s^{i,m})$, then L checks whether $f(X^i) = (i,x^1,b^1,\dots,x^m,b^m)$. Let $i^* \in [\sigma]$ be the minimum integer i for which A succeeds in inverting (if not, A returns $\perp$ and halts). Then, L outputs $(\Pi_f^{i^*})_{[i^*]}$ as a hypothesis. It is not hard to verify that L halts in time $\mathsf{poly}(n,m,\delta^{-1},t,\sigma,\tau)$.

For the correctness of L, we first show the following claim.

Claim 5.1.5. *For any polynomials $t(n), \sigma(n)$, any $t(n)$-samplable distribution $\mathcal{D} = \{\mathcal{D}_n\}_{n\in\mathbb{N}}$ on samplers, any sufficiently large $n \in \mathbb{N}$, and any $m, \tau \in \mathbb{N}$, if $\mathcal{D}$ satisfies that for all $\gamma^{-1} \in \mathbb{N}$,*

$$\Pr_{X=(\boldsymbol{x},\boldsymbol{b})\sim\mathcal{D}_n^m}\left[\mathrm{LT}^\tau(X) \le \min\{-\log \mathcal{D}_n^m(\boldsymbol{b}|\boldsymbol{x}) + C(\log\tau + \log\gamma^{-1}), \sigma(n)\}\right] \ge 1-\gamma,$$

then it holds that, for every $\gamma^{-1} \in \mathbb{N}$,

$$\Pr_{X=\{(x^i,b^i)\}_{i=1}^m\sim\mathcal{D}_n^m}\left[\Pr_{U_r}\left[f(U_r) = (\mathrm{LT}^\tau(X), x^1, b^1, \dots, x^m, b^m)\right] \ge \frac{\gamma^C}{n^2\sigma(n)\tau^C}\cdot\mathcal{D}_n^m(X)\right] \ge 1-\gamma,$$

where $r := r(n, m, t(n), \sigma(n), \tau)$.

Proof. We define a $t(n)$-samplable distribution $\mathcal{D}' = \{\mathcal{D}'_n\}_{n\in\mathbb{N}}$ such that each $\mathcal{D}'_n$ is a distribution of the first half of $S \sim \mathcal{D}_n$ (i.e., the sub-circuit that produces examples). Let M be the deterministic $t(n)$-time sampling algorithm for $\mathcal{D}'$, and let $d = |M|$.

We only consider a sufficiently large $n \in \mathbb{N}$ so that $n \ge 2^d$. Then, with probability $n^{-1}\cdot 2^{-d} \ge n^{-2}$ over the choice of $j \in [n]$ and $\Pi_e \in \{0,1\}^n$, the program $(\Pi_e)_{[j]}$ (in the input of f) corresponds to the description of M. Under this condition, the distribution of S in the computation of f is statistically identical to $\mathcal{D}'_n$, and for each $m \in \mathbb{N}$ and $(\boldsymbol{x}, \boldsymbol{b}) \in \mathrm{supp}(\mathcal{D}_n^m)$, the probability that $\boldsymbol{x}$ is sampled according to $\mathcal{D}_n^m$ (we denote this probability by $\mathcal{D}_n^m(\boldsymbol{x})$) is equivalent to the conditional probability that $\boldsymbol{x} = (x^1, \dots, x^m)$ holds for $(i, x^1, b^1, \dots, x^m, b^m) \sim f(U_r)$. Thus, we have that for each $m \in \mathbb{N}$ and each $(\boldsymbol{x}, \boldsymbol{b}) \in \mathrm{supp}(\mathcal{D}_n^m)$,

$$\Pr_{U_r}\left[\boldsymbol{x} = (x^1, \dots, x^m) \text{ for } f(U_r) = (i, x^1, b^1, \dots, x^m, b^m)\right] \ge \frac{\mathcal{D}_n^m(\boldsymbol{x})}{n^2}.$$

Fix $\gamma^{-1} \in \mathbb{N}$ and $X = (\boldsymbol{x}, \boldsymbol{b}) \in \mathrm{supp}(\mathcal{D}_n^m)$ satisfying the following condition arbitrarily:

$$\mathrm{LT}^\tau(X) \le \min\{-\log \mathcal{D}_n^m(\boldsymbol{b}|\boldsymbol{x}) + C(\log\tau + \log\gamma^{-1}), \sigma(n)\} \tag{5.1}$$

Over the choice of $i \in [\sigma(n)]$ (in the input of f), the event that $i = \mathrm{LT}^\tau(X)$ ($\le \sigma(n)$) occurs with probability $\sigma(n)^{-1}$. We consider the event $E_{\boldsymbol{x}}$ that $i = \mathrm{LT}^\tau(X)$ and $\boldsymbol{x} = (x^1, \dots, x^m)$ holds for $f(U_r) = (i, x^1, b^1, \dots, x^m, b^m)$. Then, we have

$$\Pr_{f(U_r)}[E_{\boldsymbol{x}}] \ge \frac{\mathcal{D}_n^m(\boldsymbol{x})}{n^2\sigma(n)}.$$

Under the condition that $E_{\boldsymbol{x}}$ occurs, the probability that $(\Pi_f)_{[i]}$ (note that $i = \mathrm{LT}^\tau(X)$) corresponds to the program Π^*_X satisfying that $|\Pi^*_X| = \mathrm{LT}^\tau(X)$ and $b^i = U^\tau(\Pi^*_X, x^i)$ for each $i \in [m]$ is

$$\begin{aligned}\Pr_{\Pi_f}\left[(\Pi_f)_{[i]} = \Pi^*_X\right] &= 2^{-\mathrm{LT}^\tau(X)}\\ &\ge 2^{\log \mathcal{D}_n^m(\boldsymbol{b}|\boldsymbol{x}) - C(\log\tau+\log\gamma^{-1})}\\ &= \frac{\gamma^C\cdot\mathcal{D}_n^m(\boldsymbol{b}|\boldsymbol{x})}{\tau^C},\end{aligned}$$

where the inequality follows from (5.1).

Therefore, for each $X = (\boldsymbol{x}, \boldsymbol{b}) \in \mathrm{supp}(\mathcal{D}_n^m)$ (let $\boldsymbol{x} = (x^1, \ldots, x^m)$ and $\boldsymbol{b} = (b^1, \ldots, b^m)$), if $(\boldsymbol{x}, \boldsymbol{b})$ satisfies (5.1), then we have

$$\begin{aligned}\Pr_{U_r}\left[f(U_r) = (\mathrm{LT}^\tau(X), x^1, b^1, \ldots, x^m, b^m)\right] &\geq \Pr_{\Pi_f}\left[(\Pi_f)_{[\mathrm{LT}^\tau(X)]} = \Pi_X^*\right] \cdot \Pr_{f(U_r)}[E_{\boldsymbol{x}}] \\ &\geq \frac{\gamma^C \cdot \mathcal{D}_n^m(\boldsymbol{b}|\boldsymbol{x})}{\tau^C} \cdot \frac{\mathcal{D}_n^m(\boldsymbol{x})}{n^2\sigma(n)} \\ &= \frac{\gamma^C}{n^2\sigma(n)\tau^C} \cdot \mathcal{D}_n^m(X).\end{aligned}$$

Since $\Pr_{(\boldsymbol{x},\boldsymbol{b})\sim\mathcal{D}_n^m}[(\boldsymbol{x}, \boldsymbol{b}) \text{ satisfies (5.1)}] \geq 1 - \gamma$ follows from the assumption, the claim holds. $\diamond$

Now, we show the correctness of $L(\text{-}; 1^{\langle n, m, \delta^{-1}, t(n), \sigma(n), \tau\rangle})$. Suppose that the error probability of L is grater than δ. Then, we derive a contradiction. For readability, we omit the parameters of L below.

By the assumption on $\mathcal{D}$ in the theorem and Claim 5.1.5, we have that

$$\Pr_{X=\{(x^i,b^i)\}_{i=1}^m\sim\mathcal{D}_n^m}\left[\Pr_{U_r}\left[f(U_r) = (\mathrm{LT}^\tau(X), x^1, b^1, \ldots, x^m, b^m)\right] \geq \frac{(\delta/2)^C}{n^2\sigma(n)\tau^C} \cdot \mathcal{D}_n^m(X)\right] \geq 1 - \delta/2.$$

Let $\mathrm{GoodSamp} \subseteq \mathrm{supp}(\mathcal{D}_n^m)$ be the set of samples $X \in \mathrm{supp}(\mathcal{D}_n^m)$ satisfying the event in the probability above, i.e., $\Pr_{X\sim\mathcal{D}_n^m}[X \in \mathrm{GoodSamp}] \geq 1 - \delta/2$, and for each $X = \{(x^i, b^i)\}_{i=1}^m \in \mathrm{GoodSamp}$,

$$\Pr_{U_r}\left[f(U_r) = (\mathrm{LT}^\tau(X), x^1, b^1, \ldots, x^m, b^m)\right] \geq \frac{(\delta/2)^C}{n^2\sigma(n)\tau^C} \cdot \mathcal{D}_n^m(X).$$

For each $X = \{(x^i, b^i)\}_{i=1}^m \in \mathrm{supp}(\mathcal{D}_n^m)$, we let F_X denote the event that $L(X)$ fails in finding a program $\Pi \in \{0,1\}^*$ such that $|\Pi| = \mathrm{LT}^\tau(X)$ and $b^i = U^\tau(\Pi, x^i)$ for each $i \in [m]$. Notice that F_X occurs only if $A(\mathrm{LT}^\tau(X), x^1, b^1, \ldots, x^m, b^m)$ fails in finding an inverse element.

By the assumption that the error probability of L is greater than δ, we have

$$\begin{aligned}\Pr_{X\sim\mathcal{D}_n^m,L}[F_X \wedge X \in \mathrm{GoodSamp}] &\geq \Pr_{X\sim\mathcal{D}_n^m,L}[F_X] - \Pr_{X\sim\mathcal{D}_n^m}[X \notin \mathrm{GoodSamp}] \\ &\geq \delta - \delta/2 = \delta/2.\end{aligned}$$

Furthermore, we obtain that

$$\begin{aligned}\delta/2 &\leq \Pr_{X\sim\mathcal{D}_n^m,L}[F_X \wedge X \in \mathrm{GoodSamp}] \\ &= \sum_{X\in\mathrm{GoodSamp}} \mathcal{D}_n^m(X) \cdot \Pr_L[F_X] \\ &\leq \sum_{X=\{(x^i,b^i)\}_{i=1}^m\in\mathrm{GoodSamp}} \mathcal{D}_n^m(X) \cdot \Pr_A[A(\mathrm{LT}^\tau(X), x^1, b^1, \ldots, x^m, b^m) \text{ fails in inverting}] \\ &\leq \sum_{X=\{(x^i,b^i)\}_{i=1}^m\in\mathrm{GoodSamp}} \frac{n^2\sigma(n)\tau^C}{(\delta/2)^C} \Pr_{U_r}\left[f(U_r) = (\mathrm{LT}^\tau(X), x^1, b^1, \ldots, x^m, b^m)\right] \\ &\qquad \cdot \Pr_A[A(\mathrm{LT}^\tau(X), x^1, b^1, \ldots, x^m, b^m) \text{ fails in inverting}]\end{aligned}$$

$$= \frac{n^2\sigma(n)\tau^C}{(\delta/2)^C} \Pr_{A,U_r}[A(f(U_r)) \text{ fails in inverting } \wedge \exists X = \{(x^i, b^i)\}_{i=1}^m \in \text{GoodSamp} \text{ s.t. } f(U_r) = (\mathrm{LT}^\tau(X), x^1, b^1, \ldots, x^m, b^m)]$$

$$\leq \frac{n^2\sigma(n)\tau^C}{(\delta/2)^C} \Pr_{A,U_r}[A(f(U_r)) \text{ fails in inverting}].$$

Therefore, the failure probability of A is at least

$$\frac{\delta}{2} \cdot \frac{(\delta/2)^C}{n^2\sigma(n)\tau^C} = 2\delta'.$$

This is contradiction because the failure probability of A is at most δ' by the choice of the parameter for A in L. □

Next, we show that the coding theorem in Lemma 5.1.4 holds if a distribution on samplers is separated into two distributions $\mathcal{E}$ and $\mathcal{F}$ over examples and a target function as in the BFKL model [BFKL93]) under the following derandomization assumption for nondeterministic circuits.

Hypothesis 5.1.6 (Pseudorandom generator against nondeterministic circuits)**.** *There exists a constant $C_0 > 0$ such that for every polynomial m, there exists a $\mathsf{poly}(m(n))$-time computable pseudorandom generator $G = \{G_n\}_{n\in\mathbb{N}}$, where $G_n \colon \{0,1\}^{C_0 \log m(n)} \to \{0,1\}^{m(n)}$ that $1/m(n)$-fools $m(n)$-size nondeterministic circuits, i.e., for every $m(n)$-size nondeterministic circuit C and every $n \in \mathbb{N}$,*

$$\left| \Pr_{z\sim\{0,1\}^{C_0 \log m(n)}}[C(G(z)) = 1] - \Pr_{w\sim\{0,1\}^{m(n)}}[C(w) = 1] \right| \leq \frac{1}{m(n)}.$$

For example, Hypothesis 5.1.6 holds if E requires exponential-size (single-valued) nondeterministic circuits almost everywhere [SU05].

For every distribution families $\mathcal{E} = \{\mathcal{E}_n\}_{n\in\mathbb{N}}$ and $\mathcal{F} = \{\mathcal{F}_n\}_{n\in\mathbb{N}}$, where each $\mathcal{E}_n$ is over $\{0,1\}^n$, and each $\mathcal{F}_n$ is over (binary representations of) functions in a concept class $\mathscr{C}_n$, we use the notation $\mathcal{D}_{\mathcal{E},\mathcal{F}}$ to refer to the following distribution of samplers: for each $n \in \mathbb{N}$, (i) select $f \sim \mathcal{F}_n$, and (ii) output a sampler $S_{\mathcal{E},f}$ that generates $(x, f(x))$ for $x \sim \mathcal{E}_n$.

Lemma 5.1.7. *If Hypothesis 5.1.6 holds, then for every polynomial-time evaluatable concept class $\mathscr{C} = \{\mathscr{C}_n\}_{n\in\mathbb{N}}$, where $\mathscr{C}_n \subseteq \{f \colon \{0,1\}^n \to \{0,1\}\}$, there exist a polynomial τ_0 and a constant $C > 0$ such that for every polynomial $t(n)$, every $t(n)$-time samplable distribution $\mathcal{E}$ over examples, and every $t(n)$-time samplable distribution $\mathcal{F}$ over $\mathscr{C}$, for every sufficiently large $n \in \mathbb{N}$, every $m, \tau \in \mathbb{N}$ with $\tau \geq \tau_0(n, m, t(n))$, and every $X = (\boldsymbol{x}, \boldsymbol{b}) \in \mathrm{supp}((\mathcal{D}_{\mathcal{E},\mathcal{F}})_n^m)$, it holds that*

$$\mathrm{LT}^\tau(X) \leq -\log(\mathcal{D}_{\mathcal{E},\mathcal{F}})_n^m(\boldsymbol{b}|\boldsymbol{x}) + C \log \tau.$$

Particularly, $\mathcal{D}_{\mathcal{E},\mathcal{F}}$ satisfies the condition in Theorem 5.1.4 for every $\tau \geq \tau_0(n, m, t(n))$ when $\sigma(n)$ in Theorem 5.1.4 is the upper bound on the length of binary representation for the class $\mathscr{C}_n$.

Proof. The proof essentially appeared in [AGMMM18; Hir21b], the difference is only that we consider the time-bounded LT-complexity.

For readability, we omit the subscript $\mathcal{E}, \mathcal{F}$ of $\mathcal{D}_{\mathcal{E},\mathcal{F}}$. Let F be the $t(n)$-time sampling algorithm for $\mathcal{F}$.

Fix $X = (\boldsymbol{x}, \boldsymbol{b}) \sim \mathcal{D}_n^m$ arbitrarily. Let $\boldsymbol{x} = (x^1, \dots, x^m)$, $\boldsymbol{b} = (b^1, \dots, b^m)$, and $p = \mathcal{D}_n^m(\boldsymbol{b}|\boldsymbol{x})$. Then, we have

$$\begin{aligned} p = \mathcal{D}_n^m(\boldsymbol{b}|\boldsymbol{x}) &= \Pr_{f \sim \mathcal{F}_n}[f(x^i) = b^i \text{ for each } i \in [m]] \\ &= \Pr_{r \sim \{0,1\}^{t(n)}}[f = F(1^n, r) \text{ and } f(x^i) = b^i \text{ for each } i \in [m]] \end{aligned}$$

Let $R \subseteq \{0,1\}^{t(n)}$ be a set of random strings $r \in \{0,1\}^{t(n)}$ such that $f = F(1^n, r)$ and $f(x^i) = b^i$ for each $i \in [m]$. Then, we have $p = |R| \cdot 2^{-t(n)}$.

Let $s = \lceil -\log p \rceil$ and $\ell = t(n) - s - 2$. We consider the pairwise-independent hash family $\mathcal{H} = \{h_{U,v} : \{0,1\}^{t(n)} \to \{0,1\}^\ell\}_{(U,v)}$, where $U \in \mathbb{F}_2^{\ell \times t(n)}$, $v \in \mathbb{F}_2^\ell$, and $h_{U,v}(r) = U \cdot r + v$ (where we identify $\{0,1\}$ with $\mathbb{F}_2$).

By Chebyshev's inequality, with probability at least $1/4$ over the choice of (U, v), there exists an element $r_{U,v} \in R \cap h_{U,v}^{-1}(0^\ell)$, and $|h_{U,v}^{-1}(0^\ell)| \leq 2^{s+3}$ holds (cf. [AGMMM18, Claim 4.2.1]). Furthermore, by Gaussian elimination, any $r \in h_{U,v}^{-1}(0^\ell)$ is reconstructed from (U, v) and additional $\log |h_{U,v}^{-1}(0^\ell)| \leq s + 3$ bits of information (i.e., index in $h_{U,v}^{-1}(0^\ell)$) in $\mathsf{poly}(t(n))$ time (cf. [AGMMM18, Claim 4.2.2]). Particularly, $r_{U,v} \in R \cap h_{U,v}^{-1}(0^\ell)$ is also reconstructed from (U, v) and some $w_{U,v} \in \{0,1\}^{\leq s+3}$.

Now, we consider the following nondeterministic algorithm A. On input $z \in \{0,1\}^{\ell(t(n)+1)}$ and auxiliary advice $\boldsymbol{x} = (x^1, \dots, x^m)$ and $\boldsymbol{b} = (b^1, \dots, b^m)$, the algorithm A regards z as a tuple (U, v), where $U \in \mathbb{F}_2^{\ell \times t(n)}$ and $v \in \mathbb{F}_2^\ell$, and nondeterministically guesses $w_z \in \{0,1\}^{\leq s+3}$ such that the reconstruction algorithm specified above generates a function $f \colon \{0,1\}^n \to \{0,1\}$ such that $f(x^i) = b^i$ holds for each $i \in [m]$. If there exists such a w_z, then A accepts z. Notice that, by the argument above, the following holds:

$$\Pr_{z \sim \{0,1\}^{\ell(t(n)+1)}}[A(z; \boldsymbol{x}, \boldsymbol{b}) = 1] \geq 1/4.$$

By the standard way to translate a Turing machine into a circuit, we obtain a nondeterministic circuit $\tilde{A}$ of size $\tau(n, m, t(n))$ that corresponds to A, where τ is a polynomial determined by the time complexity for evaluating $\mathscr{C}$ (because A needs to execute the evaluation algorithm for $\mathscr{C}$). Let $G \colon \{0,1\}^{C_0 \log \tau(n,m,t(n))} \to \{0,1\}^{\tau(n,m,t(n))}$ be the pseudorandom generator in Hypothesis 5.1.6 for circuit size $\tau(n, m, t(n))$. Then, there must exist a string $z' \in \{0,1\}^{C_0 \log \tau(n,m,t(n))}$ such that $\tilde{A}(G(z')) = 1$; otherwise,

$$\Pr_{z \sim \{0,1\}^{\tau(n,m,t(n))}}[\tilde{A}(z) = 1] - \Pr_{z' \sim \{0,1\}^{C_0 \log \tau(n,m,t(n))}}[\tilde{A}(G(z')) = 1] \geq 1/4 - 0 = 1/4,$$

which contradicts that G is a pseudorandom generator.

Since $\tilde{A}(G(z')) = 1$, there exists a witness $w \in \{0,1\}^{\leq s+3}$ such that the reconstruction algorithm above generates $f \colon \{0,1\}^n \to \{0,1\}$ satisfying that $f(x^i) = b^i$ holds for each $i \in [m]$ from $G(z')$ (regarded as a seed for the hash function) and index w. We remark that this consistent function f is uniformly constructed from G, z', w in $\tau'(n, m, t(n))$ time, where τ' is a polynomial determined by the time-complexity of evaluating $\mathscr{C}$ and computing G.

Thus, there exists a polynomial τ'' (determined by the time-complexity of evaluating $\mathscr{C}$ and

computing G, and simulation overhead for U) such that

$$\begin{aligned}\mathrm{LT}^{\tau''(n,m,t(n))}(X) &\leq |G| + |z'| + |w| + O(1)\\ &\leq s + O(\log \tau''(n,m,t(n)))\\ &\leq -\log \mathcal{D}_n^m(\boldsymbol{b}|\boldsymbol{x}) + C_1 \log \tau''(n,m,t(n)),\end{aligned}$$

for some absolute constant $C_1 > 0$. □

Lemmas 5.1.4 and 5.1.7 immediately imply the average-case easiness of MINLT in the BFKL model under the non-existence of OWF.

Corollary 5.1.8 (Item 1 ⇒ Item 2 in Theorem 5.1.3)**.** *Let $\mathscr{C} = \{\mathscr{C}_n\}_{n\in\mathbb{N}}$ be a polynomial-time evaluatable class, where $\mathscr{C}_n \subseteq \{f\colon \{0,1\}^n \to \{0,1\}\}$, such that the length of binary representation for $\mathscr{C}_n$ is at most $\ell_{\mathscr{C}}(n)$ for some polynomial $\ell_{\mathscr{C}}$.*

Under the non-existence of infinitely-often one-way functions and Hypothesis 5.1.6, there exist a polynomial-time randomized algorithm L and a polynomial τ_0 such that for every polynomial $t(n)$, every unknown $t(n)$-time samplable distribution $\mathcal{E}$ over examples, every unknown $t(n)$-time samplable distribution $\mathcal{F}$ over $\mathscr{C}$, every sufficiently large $n \in \mathbb{N}$, and every $m, \delta^{-1}, \tau \in \mathbb{N}$ with $\tau \geq \tau_0(n,m,t(n))$, the algorithm $L(\text{-}; 1^{\langle n,m,\delta^{-1},t(n),\ell_{\mathscr{C}}(n),\tau\rangle})$ solves the search version of MINLT$[\tau]$ *on average under $\mathcal{D}^m_{\mathcal{E},\mathcal{F}}$ with error probability at most δ.*

5.2 Learning vs. Auxiliary-Input One-Way Functions

In this section, we present the characterization of the existence of AIOWFs by the hardness of learning in the hybrid model between the PAC learning model and the BFKL model, which is the same as the BFKL model except that we consider arbitrary (possibly not efficiently samplable) example distributions as the original PAC learning model.

Theorem 5.2.1. *The following are equivalent:*

1. *There exists no AIOWF;*
2. *Every polynomial-time-evaluatable concept class $\mathscr{C}$ and every sampleable distribution $\mathcal{F}$ over $\mathscr{C}$, the class $\mathscr{C}$ is PAC learnable on average in the BFKL model with respect to all unknown example distributions and $\mathcal{F}$;*
3. *Every polynomial-time-evaluatable concept class $\mathscr{C}$ and every sampleable distribution $\mathcal{F}$ over $\mathscr{C}$, the class $\mathscr{C}$ is weakly PAC learnable on average in the BFKL model with respect to all unknown example distributions and $\mathcal{F}$.*

Note that Theorem 5.2.1 yields several new insights into the hardness of PAC learning. For instance, Vadhan [Vad06] showed the characterization of computational zero-knowledge in cryptography by AIOWF. Thus, our result provides a closer relationship between the average-case hardness of learning and the hardness of obtaining knowledge, as studied in cryptography. Even within learning theory, this result implies that if *worst-case* PAC learning *on the uniform distribution* is easy, then *average-case* PAC learning *on any distribution* is also easy because the former is sufficient to break any auxiliary-input cryptography, as observed by Applebaum, Barak, and Xiao [ABX08]. Such a relationship seems quite non-trivial and fundamental knowledge on two worst-case requirements in PAC learning (on target functions and example distributions), which had not been known before.

5.2.1 Intuition: Duality between Learning and Auxiliary-Input Cryptography

We give an intuition of the correspondence between hardness of learning and auxiliary-input one-way functions. The essential idea is same as [BFKL93] except the consideration of auxiliary-input.

Let $\mathscr{C}$ be a concept class. As discussed in Chapter 2.6, the essential task of PAC learning $\mathscr{C}$ is, for a given sample set $(x_1, b_1), \ldots, (x_m, b_m)$, to find a consistent function $f \in \mathscr{C}$ such that $f(x^i) = b^i$ for each i by Occam's razor [BEHW87].

Now, we regard the examples $x_1, \ldots, x_m$ as auxiliary-input and the hidden function f as a hidden random seed. Then, we can regard the labels $b_1, \ldots, b_m$ as the output of such an auxiliary-input function. Additionally if the hidden function f is selected uniformly at random, it is not hard to verify that the task of finding a consistent function exactly corresponds to the task of inverting AIOWF. Even if f is selected according to some samplable distribution $\mathcal{F}$, the same argument holds by considering the random seed for $\mathcal{F}$ as a random seed for AIOWF.

We remark that, in the simple outline above, the roles of inputs and functions are switched in learning and auxiliary-input cryptography. In learning, the public information is examples (i.e., input for a target function) and the secret information is a target function. By contrast, in auxiliary-input cryptography, the public information is the auxiliary-input (i.e., the description of cryptographic primitives) and the secret information is a random seed (i.e., input for cryptographic primitives). This correspondence clearly explains why the hardness of learning with the *worst-case* requirement on example distributions corresponds to the existence of AIOWF with the *worst-case* requirement on auxiliary-input. Furthermore, the idea of switching the roles of inputs and functions is crucial for the duality of learning and cryptography and applied many times in the remainder of the thesis. Particularly, a similar technique is used in Chapter 7 to obtain the characterization of the hardness of standard PAC learning by another auxiliary-input cryptographic primitive. In addition, switching of the roles of inputs and functions will be further studied in a more fine-grained way in Chapter 6, which is at the core of the first characterization of polynomial-stretch pseudorandom generators computable in constant parallel time (i.e., NC^0).

5.2.2 Proof of Theorem 5.2.1

First, we show the implication from the non-existence of AIOWF to the feasibility of learning. We use the following proposition.

Proposition 5.2.2. *If there exists no auxiliary-input one-way function, then for every polynomial-time-computable auxiliary-input function $f = \left\{f_z \colon \{0,1\}^{s(|z|)} \to \{0,1\}^{t(|z|)}\right\}_{z\in\{0,1\}^*}$, there exists a randomized polynomial-time algorithm A such that for every $z \in \{0,1\}^*$ and every $\delta^{-1} \in \mathbb{N}$,*

$$\Pr_{x\sim\{0,1\}^{s(|z|)},A}[A(z, f_z(x); 1^{\delta^{-1}}) \notin f_z^{-1}(f_z(x))] \leq \delta.$$

Proof Sketch. We define a new family $f' = \left\{f'_{z,k}\right\}_{z\in\{0,1\}^*,k\in\mathbb{N}}$ so that

$$f'_{z,k}(x_1, \ldots, x_k) = (f_z(x_1), \ldots, f_z(x_k))$$

for every $x_1, \ldots, x_k \in \{0,1\}^{s(n)}$, where we implicitly assume that (z, k) is encoded into a binary string. Yao's amplification theorem [Yao82; Gol01] shows that inverting f_z with probability $1 - \delta$ reduces to the task of inverting $f_{z,k}$ for some $k = \mathsf{poly}(|z|, \delta^{-1})$. □

Lemma 5.2.3 (Item 1 ⇒ Item 2 in Theorem 5.2.1)**.** *There exists no AIOWF, then every polynomial-time-evaluatable concept class $\mathscr{C}$ and every sampleable distribution $\mathcal{F}$ over $\mathscr{C}$, the class $\mathscr{C}$ is PAC learnable on average in the BFKL model with respect to all unknown example distributions and $\mathcal{F}$.*

Proof. Let $\mathscr{C}$ be an arbitrary polynomial-time-evaluatable concept class. Note that, to show Lemma 5.2.3, we only need to consider the uniform distribution over functions in $\mathscr{C}$ as the samplable distribution $\mathcal{F}$ over $\mathscr{C}$ because we can also contain the sampler F of $\mathcal{F}$ in the evaluation algorithm for $\mathscr{C}$ and regard the uniformly random seed for F as the representation of the concept class. By this argument, we can also assume the set of representations for $\mathscr{C}_n$ exactly corresponds to $\{0,1\}^{s(n)}$, where s is a polynomial. Let C be a polynomial-time evaluation algorithm for $\mathscr{C}$.

We construct a polynomial-time computable auxiliary-input function $g = \{g_{n,m,z} \colon \{0,1\}^{s(n)} \to \{0,1\}^m\}_{n,m\in\mathbb{N},z\in\{0,1\}^{nm}}$ (where we implicitly assume that (n,m,z) is encoded into a binary string) as follows: For each $n, m \in \mathbb{N}$ and $z = x^1 \circ \cdots \circ x^m$ (where $x^i \in \{0,1\}^n$ for each i),

$$g_{n,m,z}(f) = f(x^1) \circ \cdots \circ f(x^m),$$

where we identify $f \sim \{0,1\}^{s(n)}$ with a function in $\mathscr{C}_n$. Note that each $f(x^i)$ is actually computed as $C(f, x^i)$. Since C is polynomial-time computable, g is also polynomial-time computable.

By Proposition 5.2.2, there exists an randomized algorithm A such that for every $n, m, \gamma^{-1} \in \mathbb{N}$ and every $x^1, \ldots, x^m \in \{0,1\}^n$,

$$\Pr_{f\sim\{0,1\}^{s(n)},A}[A(n,m,z,g_{n,m,z}(f);1^{\gamma^{-1}}) \notin g^{-1}_{n,m,z}(g_{n,m,z}(f))] \le \gamma, \tag{5.2}$$

where $z := x^1 \circ \cdots \circ x^m$.

We construct the learner L as follows: On input $1^n, 1^\epsilon, 1^\delta$ and given access to $\mathsf{EX}_{f,\mathcal{D}}$, the learner L collects $m := \lceil \epsilon^{-1}(s(n) + \ln \delta^{-1} + 1)\rceil$ samples $(x^1, b^1), \ldots, (x^m, b^m)$ from $\mathsf{EX}_{f,\mathcal{D}}$ and executes $\tilde{f} \leftarrow A(n, m, x^1 \circ \cdots \circ x^m, b^1 \circ \cdots \circ b^m; 1^{2\delta^{-1}})$. If A succeeds in inverting (it is easily verified), then L outputs $\tilde{f}$ as a hypothesis (otherwise, L outputs $\bot$). It is easy to verify that L halts in polynomial time in n, ϵ^{-1} and δ^{-1}.

We verify the correctness. Fix $n, \epsilon^{-1}, \delta^{-1} \in \mathbb{N}$ arbitrarily, and let $\mathcal{D}$ be an arbitrary example distribution over $\{0,1\}^n$. Let $m := \lceil \epsilon^{-1}(s(n) + \ln \delta^{-1} + 1)\rceil$. For convenience, we may regard the m samples is given as input for L. By Eq. (5.2), we have

$$\Pr_{x^1,\ldots,x^m\sim\mathcal{D},L,f}\left[L(1^n,1^\epsilon,1^\delta,(x^1,b^1),\ldots,(x^m,b^m)) \text{ outputs some hypothesis } \tilde{f} \neq \bot\right] \ge 1 - \delta/2.$$

Notice that whenever $L(1^n, 1^\epsilon, 1^\delta, (x^1, b^1), \ldots, (x^m, b^m))$ outputs $\tilde{f} \neq \bot$, it holds that $\tilde{f}(x^i) = b^i$ for each $i \in [m]$. In fact, this is sufficient to apply the Occam's razor [BEHW87]. For completeness, we present the formal proof. Trivially, it suffices to show the following claim.

Claim 5.2.4.

$$\Pr_{f,\mathsf{EX},L}\left[L^{\mathsf{EX}}(1^n, 1^{\epsilon^{-1}}, 1^{\delta^{-1}}) \textit{ outputs some hypothesis } \tilde{f} \neq \bot \textit{ and } \Pr_{x\sim\mathcal{D}}[\tilde{f}(x) \neq f(x)] > \epsilon\right] \le \delta/2.$$

Proof. We define a subset $B_{\mathcal{D}}(f,\epsilon) \subseteq \{0,1\}^{s(n)}$ as the set of representations of functions in $\mathscr{C}_n$ that are ϵ-far from f under $\mathcal{D}$, i.e.,

$$B_{\mathcal{D}}(f,\epsilon) = \left\{\tilde{f} \in \{0,1\}^{s(n)} : \Pr_{x\sim\mathcal{D}}[\tilde{f}(x) \neq f(x)] > \epsilon\right\}.$$

The probability in the claim is bounded above by

$$\mathop{\mathbb{E}}_{f}\left[\sum_{\tilde{f}\in\{0,1\}^{s(n)}}\Pr_{L,\mathsf{EX}}[L^{\mathsf{EX}}\text{ outputs }\tilde{f}\wedge\tilde{f}\in B_{\mathcal{D}}(f,\epsilon)]\right]=\mathop{\mathbb{E}}_{f}\left[\sum_{\tilde{f}\in B_{\mathcal{D}}(f,\epsilon)}\Pr_{L,\mathsf{EX}}[L^{\mathsf{EX}}\text{ outputs }\tilde{f}]\right]$$

Whenever L outputs $\tilde{f}\neq\bot$, the hypothesis $\tilde{f}$ is consistent with the target function on the given samples (otherwise, the inverter fails, and L outputs $\bot$). Thus, the expectation is bounded above by

$$\begin{aligned}
\mathop{\mathbb{E}}_{f}\left[\sum_{\tilde{f}\in B_{\mathcal{D}}(f,\epsilon)}\Pr_{x^1,\ldots,x^m\sim\mathcal{D}}[\tilde{f}(x^i)=f(x^i)\text{ for all }i\in[m]]\right]&=\mathop{\mathbb{E}}_{f}\left[\sum_{\tilde{f}\in B_{\mathcal{D}}(f,\epsilon)}\left(\Pr_{x\sim\mathcal{D}_n}[\tilde{f}(x)=f(x)]\right)^m\right]\\
&\leq\mathop{\mathbb{E}}_{f}\left[\sum_{\tilde{f}\in B_{\mathcal{D}}(f,\epsilon)}(1-\epsilon)^m\right]\\
&\leq\mathop{\mathbb{E}}_{f}\left[|B_{\mathcal{D}}(f,\epsilon)|(1-\epsilon)^m\right]\\
&\leq\mathop{\mathbb{E}}_{f}\left[2^{s(n)}(1-\epsilon)^m\right]\\
&=2^{s(n)}(1-\epsilon)^m\\
&\leq 2^{s(n)}(1-\epsilon)^{\epsilon^{-1}(s(n)+\ln\delta^{-1}+1)}\\
&\leq 2^{s(n)}/e^{(s(n)+\ln\delta^{-1}+1)}\\
&<\delta/2,
\end{aligned}$$

where the first inequality follows from the definition of $B_{\mathcal{D}}(f,\epsilon)$. ◇

□

Next, we show that the other direction, i.e., the implication from the feasibility of learning to the non-existence of auxiliary-input one-way functions.

Lemma 5.2.5 (Item 3 ⇒ Item 1 in Theorem 5.2.1)**.** *If every polynomial-time-evaluatable concept class $\mathscr{C}$ and every sampleable distribution $\mathcal{F}$ over $\mathscr{C}$, the class $\mathscr{C}$ is PAC learnable on average in the BFKL model with respect to all unknown example distributions and $\mathcal{F}$, then there exists no AIPRF (thus there exists no AIOWF by Theorem 2.5.8).*

Note that this lemma completes the proof of Theorem 5.2.1: (3 ⇒ 1) follows from the lemma above; (1 ⇒ 2) follows from Lemma 5.2.3; and (2 ⇒ 3) trivially holds by definition.

Proof. Suppose for contradiction that AIPRF $F=\{F_z:\{0,1\}^{|z|}\times\{0,1\}^{|z|}\to\{0,1\}\}_{z\in\{0,1\}^*}$ exists.

We define a concept class $\mathscr{C}$ specified by the following evaluation algorithm $C=\{C_n:\{0,1\}^{\lceil n/2\rceil}\times\{0,1\}^n\to\{0,1\}\}_{n\in\mathbb{N}}$: for each $n\in\mathbb{N}$,

$$\begin{aligned}
C_{2n-1}(u,y)&=1\\
C_{2n}(u,z\circ x)&=F_z(u,x),
\end{aligned}$$

where $u \in \{0,1\}^n$, $z, x \in \{0,1\}^n$, and $y \in \{0,1\}^{2n-1}$. It is easily verified that $\mathscr{C}$ is evaluated in polynomial time.

For convenience, we consider the average-case weak prediction model in Definition 2.3.4. By the assumption, there exists a weak learner L that learns $\mathscr{C}$ in the BFKL model with advantage $\mathsf{poly}^{-1}(|z|)$, where we consider the uniform distribution over functions in $\mathscr{C}$ (i.e., the uniform distribution over u in the above) and an arbitrary unknown example distribution.

We construct an adversary A for F based on L as follows: On input $z \in \{0,1\}^n$ (i.e., auxiliary-input of F_z) and query access to a function $f : \{0,1\}^n \to \{0,1\}$ (which is either of a pseudorandom function or a truly random function), A executes L, where A passes $(z \circ x, f(x))$ as each example and $z \circ x^*$ as a challenge for independently selected $x, x^* \sim \{0,1\}^n$. If L outputs some prediction b, then A checks whether $f(x^*) = b$ by its own oracle access. If L succeeds in predicting, A outputs 1, otherwise, it outputs 0. Notice that the probability that A outputs 1 is exactly the same as the probability that L succeeds in predicting over the choice of f and randomness for A. Obviously, A halts in polynomial time in n.

On one hand, consider the case where f is selected from pseudorandom functions. In this case, it is not hard to verify that A executes P in the valid setting where the target function is f and the example distribution is $z \circ U_n$. Thus, A outputs 1 with probability at least $1/2 + 1/\mathsf{poly}(n)$.

On the other hand, consider the case where f is selected from all n-input functions, i.e., a truly random function. Because P looks at only $\mathsf{poly}(n)$ values of f, randomly selected x^* is contained in previous examples with only negligibly small probability. If x^* is not contained in the previous examples, then P cannot guess the value of $f(x^*)$ better than at random because f is a truly random function. Thus, A outputs 1 with probability at most $1/2 + \mathsf{negl}(n)$.

Therefore, A distinguishes the above two cases with non-negligible probability, which contradicts the assumption that F is AIPRF. □

Chapter 6

Learning versus Pseudorandom Generators in Constant Parallel Time

In this chapter, we study relationships between learning and pseudorandomness in extremely low complexity regimes. Particularly, we show the first characterization result of an important cryptographic primitive, *polynomial-stretch pseudorandom generators in constant parallel time*, by average-case hardness of natural learning problems.

6.1 Background

A dichotomy between learning and cryptography is one of the central topics in theoretical computer science. An implication from cryptography to hardness of learning has already been studied in the pioneering work by Valiant [Val84], who observed that the existence of a secure cryptographic primitive implies the hardness of learning polynomial-size circuits ($\mathsf{P/poly}$). Many follow-up studies further showed the hardness of learning more restricted classes such as AC^0 under several cryptographic or deeply related assumptions [KV94a; Kha93; AK95; ABW10; Dan16; DS16; Vad17; DV21]. The opposite implication from hardness of learning to cryptography is relatively less understood and first studied by Impagliazzo and Levin [IL90] and Blum, Furst, Kearns, and Lipton [BFKL93]. Particularly, Blum, Furst, Kearns, and Lipton [BFKL93] formulated the average-case hardness of PAC learning and presented constructions of several cryptographic primitives based on the average-case hardness of learning. These early studies characterized a central cryptographic primitive called a one-way function (OWF) by the average-case hardness of learning $\mathsf{P/poly}$. The dichotomy between learning and cryptography has been further studied over decades in various settings [NR06; OS17; San20]. In previous chapters, we also presented several equivalence results between the average-case hardness of learning polynomial-time evaluated classes and general polynomial-time computable cryptographic primitives.

In general, the complexity for computing cryptographic primitives is deeply related to the complexity of a concept class for learning (i.e., a class of target functions learners try to learn). This observation leads us to study the dichotomy between learning and cryptography in low complexity classes. One motivation of this is highly efficient cryptography based on the hardness assumption of learning simple classes, as mentioned by Blum, Furst, Kearns, and Lipton [BFKL93]. This direction is successful in certain fields; e.g., several candidates for a cryptographic primitive called a weak pseudorandom function were proposed in low complexity based on the hardness of learning

problems for which no efficient algorithm is known at present [ABGKR14; BCGIKS21]. Another motivation is identifying the capability of efficient learning based on well-established arguments in cryptography. This direction has also been demonstrated for decades in studies on cryptographic hardness of learning [e.g., KV94a; Kha93; AK95; AR16; DV21].

In this work, we study a dichotomy between learning and polynomial-stretch pseudorandom generators (PPRGs) computable in constant-depth circuits (i.e., NC^0), where a PPRG is a fundamental cryptographic primitive stretching a given n-bit random seed into an $n^{1+\Theta(1)}$-bit pseudorandom string that is indistinguishable from a truly random string by efficient adversaries. This research question is strongly motivated by both sides of constructing highly efficient cryptography and identifying the capability of efficient learning. Below, we explain further backgrounds.

From the perspective of cryptography. A PPRG in NC^0 is one of the most studied primitives in parallel cryptography [cf. CM01; App13] because of its remarkable applications, such as highly efficient cryptography [IKOS08] and a recent breakthrough on indistinguishability obfuscation ($i\mathcal{O}$) based on well-founded assumptions [JLS21; JLS22]. Despite its importance, to the best of our knowledge, the only known framework for constructing PPRGs in NC^0 is one based on Goldreich's OWF [Gol11a]. For example, the celebrated work by Applebaum, Ishai, and Kushilevitz [AIK06] on randomized encodings only yields the characterization of *sublinear-stretch* PRGs in NC^0, but characterizing PPRGs in NC^0 seems out of reach at present. Therefore, it is natural to inquire into a new candidate for PPRGs in NC^0 and a characterization result through the lens of the dichotomy between learning and cryptography.

Strictly speaking, we mainly discuss a generator defined as a collection of PPRGs, where the generator has a public index randomly and efficiently (but not in NC^0) selected in the preprocessing [cf. Gol01, Section 2.4.2]. This relaxed setting is standard, especially when we discuss a PPRG in NC^0 [cf. App13, Remark 1.1], and such relaxation does not affect the applications mentioned above.

From the perspective of computational learning theory. An ultimate goal in computational learning theory is to identify the *simplest* concept class that is not efficiently learnable under a plausible hardness assumption. Many hardness results of learning in the current frontline are related to PPRGs in NC^0. Applebaum, Barak, and Wigderson [ABW10] proved the hardness of learning $O(\log n)$-junta functions under the existence of PPRGs in NC^0 with an additional assumption on input-output connections. Applebaum and Raykov [AR16] and Daniely and Vardi [DV21] proved the hardness of learning for central classes such as depth-3 AC^0 circuits and $\omega(1)$-term DNF formulas under assumptions related to polynomial-stretch Goldreich's PRG, which is a special case where the output bits are computed by one fixed predicate. Oliveira, Santhanam, and Tell [OST22] proved that a security of polynomial-stretch Goldreich's PRG implies the impossibility of improving parameters of natural properties for simple classes such as DNF-XOR circuits under a plausible assumptions on the existence of suitable expanders, where a natural property is a notion deeply related to learning [CIKK16; CIKK17].

Since the equivalence between pseudorandomness and unpredictability follows from the well-known result by Yao [Yao82], a reader might expect a correspondence between PPRG in NC^0 and hardness of learning NC^0. However, this intuition seems incorrect because while a PPRG in NC^0 is conjectured to exist, learning NC^0 (i.e., functions with constant locality) is trivially feasible by applying Occam's razor [BEHW87]. In this sense, there seems to exist a gap between pseudorandomness and hardness of learning when we consider considerably low complexity classes such as NC^0. Nevertheless, can we obtain some learning-theoretic characterization for a collection

of PPRG in NC^0? In this work, we provide an affirmative answer to this question.

6.1.1 Our Learning Model

We introduce the learning model mainly discussed in this work, which is a natural variant of the PAC learning model. For the formal definition, see Section 6.5.2.

In this chapter, we always consider the prediction version of the BFKL model [BFKL93] and do not mention it explicitly. Remember that, in the prediction version of the BFKL model, an unknown Boolean-valued target function f (contained in some concept class $\mathscr{C}$) is selected according to a known *target distribution*[1], and a learner is given samples of the form $(x, f(x))$, where x is an example and selected identically and independently according to a known *example distribution*. After learning with the samples, the learner tries to guess a value of $f(x)$ for an additionally given input x (i.e., challenge) selected according to the same example distribution with good probability; specifically, with probability at least $1/2 + \gamma$ (we refer to γ as an *advantage*) over the choices of randomness for the learner, samples, and a target function.

A new perspective in this paper is to consider parameterized complexity of learning for a parameterized concept class and parameterized classes of example distributions and target distributions. We remark that parameterized learnability has been discussed in certain previous studies [e.g. AKL09]. The main difference from the previous formulation is the separate consideration of time complexity and sample complexity. In this paper, we only consider fixed-parameter tractability on sample complexity, and the time complexity can be arbitrary polynomial depending on parameters (or sub-exponential functions). Specifically, for parameters $k_1, \ldots, k_c$ on a concept class $\mathscr{C}$ and classes of example distributions and target distributions, we say that $\mathscr{C}$ is learnable with $(k_1, \ldots, k_c)$-FPT samples if $\mathscr{C}$ is learnable with $f(k_1, \ldots, k_c) \cdot n^{\Theta(1)}$ samples, where f is some computable function. Our learning model captures a (natural) situation in which collecting labeled data is more expensive than using computational resources. This formulation also provides a new perspective on parameterized complexity of learning; e.g., PAC learning k-junta (i.e., functions depending on only k coordinates of the input) is known to be W[2]-hard [AKL09], but feasible with FPT samples (with $k2^k \cdot n^{\Theta(1)}$ samples and in $O(n^k)$ time) by Occam's razor [BEHW87]. By contrast, it can be shown that learning degree-d $\mathbb{F}_2$-polynomials is infeasible even in this setting based on the VC theory [cf. SB14].[2]

We define the sparsity of a distribution as the maximum Hamming weight of samples.

Definition 6.1.1. *For $c \in \mathbb{N}$, we say that a family $\mathcal{D} = \{\mathcal{D}_n\}_{n\in\mathbb{N}}$ of distributions on $\{0,1\}^*$ is c-sparse if $\Pr_{x\leftarrow\mathcal{D}_n}[wt(x) \leq c] \geq 1 - \mathsf{negl}(n)$, where $wt(x)$ represents the Hamming weight of x, and $\mathsf{negl}(n)$ represents some negligible function, i.e., for any polynomial $p(n)$, it holds that $\mathsf{negl}(n) < 1/p(n)$ for any sufficiently large $n \in \mathbb{N}$.*

6.2 Our Results

As a main result, we show that a collection of PPRGs in NC^0 is characterized by the learnability of various central classes with FPT samples with respect to a sparse example distribution and an NC^0-samplable target distribution.

[1] In this chapter, we use the term "target distribution" to refer to the distribution over target functions for convenience.

[2] We can also show that learning degree-d $\mathbb{F}_2$-polynomials with FPT samples is infeasible even in the *average-case* setting over uniformly random degree-d $\mathbb{F}_2$-polynomials (see Lemma 6.7.3).

Theorem 6.2.1 (informal)**.** *The following are equivalent:*

1. *There exists a collection of (infinitely-often secure) PPRGs in* NC^0*.*

2. *c-sparse* $\mathbb{F}_2$*-polynomials are not polynomial-time learnable on average with respect to a target distribution samplable by a depth-d* NC^0 *circuit and a samplable distribution on* c'*-sparse example distributions with* (c, c', d)*-FPT samples.*

3. *c-Fourier-sparse functions are not polynomial-time learnable on average with respect to a target distribution samplable by a depth-d* NC^0 *circuit and a samplable distribution on* c'*-sparse example distributions with* (c, c', d)*-FPT samples.*

4. *For any* $f \in \{\mathrm{OR}\} \cup \{\mathrm{MOD}_m : m \in \mathbb{N} \setminus \{1\}\}$*, depth-d* f*-decision trees are not polynomial-time learnable on average with respect to a target distribution samplable by a depth-*d' NC^0 *circuit and a samplable distribution on c-sparse example distributions with* (d, c, d')*-FPT samples.*

Informally, Theorem 6.2.1 yields a new dichotomy between highly efficient pseudorandom generators and sample-efficient heuristics for learning with sparse data. Below we argue that the learning settings of Theorem 6.2.1 are natural.

Concept classes. For the formal descriptions of each parameterized concept class, see Section 6.5.1. Here, we remark that the sparsity of $\mathbb{F}_2$-polynomials and Fourier representations is one of the most important complexities of Boolean functions [cf. ODo14]. The fourth item above concerns the extensions of decision trees, containing the well-studied class of parity decision trees[3] [e.g., KM93]. Although OR decision trees have received relatively less research attention compared with the other concepts, learning OR decision trees with sparse data seems to be a natural setting where the decision is made by a few queries about whether some unusual features are observed. Interestingly, our result shows that the average-case learnability for these various concepts becomes equivalent when data are sparse through the existence of a collection of PPRGs in NC^0.

Example distributions. We remark two points. First, we consider a *distribution of* example distributions (i.e., average cases on example distributions), where the example distribution is selected at the initialization step (see Definition 6.5.5 for the formal description). Note that this captures more general settings of learning than the original BFKL model [BFKL93]; e.g., our framework captures a distribution determined by some hidden random parameter. From the perspective of cryptography, the hardness assumption on a distribution of example distributions is weaker than ones in distribution-specific settings. Second, we consider learning on sparse example distributions. Such a learning framework naturally captures learning on data with rarely observed features, such as symptoms of patients.

Target distributions. We consider NC^0-samplable distributions as target distributions, and this is a natural assumption in average-case complexity theory in learning; e.g., the uniform distribution over functions in $\mathscr{C}$ is often regarded as a projection of random strings onto the binary representations for functions in $\mathscr{C}$ (e.g., random DNFs), and almost all target distributions considered in previous studies on average-case learning are NC^0-samplable [JS05; Sel08; Sel09; JLSW11; AC15].

[3] In fact, the equivalence between constant-depth parity decision trees and constant-Fourier-sparse functions follows from the work by Kushilevitz and Mansour [KM93]. However, it is unclear whether these learning settings are equivalent when we restrict the target distributions to NC^0-samplable because the transformation between these representations may be infeasible in NC^0.

We also remark that Theorem 6.2.1 holds even in super-polynomial regimes; e.g., sub-exponential-time average-case hardness of learning with FPT samples corresponds to a collection of PPRGs secure against sub-exponential-time adversaries (where the loss of security is only polynomial). Note that super-polynomial security is applied for the construction of $i\mathcal{O}$ based on well-founded assumptions [JLS21; JLS22]. Particularly, Jain, Lin, and Sahai [JLS21] assumed (i) the hardness of learning problems LWE and LPN, (ii) the existence of a collection of PPRGs in NC^0, and (iii) the Diffie-Hellman-style assumption (i.e., SXDH). Our result characterizes assumption (ii) based on the hardness of learning and, along with their work, opens an interesting research direction: Is the well-founded hardness assumption of learning sufficient for constructing $i\mathcal{O}$ (i.e., Obfustopia)?

Next, we present several related results on the hardness of learning and PPRGs in relaxed complexity classes, which are obtained by relaxing some conditions in Theorem 6.2.1.

On removing sparsity conditions. Although Theorem 6.2.1 shows one characterization of a collection of PPRGs in NC^0 by learnability with sparse data, the sparsity is somewhat restrictive, and there exist a large amount of non-sparse data in the real world. As a second result, we show that learnability with non-sparse data for the classes in Theorem 6.2.1 still characterizes a collection of PPRGs in superclasses of NC^0.

Theorem 6.2.2 (informal)**.** *The following hold:*

1. *There exists a collection of PPRGs in $O(1)$-sparse $\mathbb{F}_2$-polynomials iff c-sparse $\mathbb{F}_2$-polynomials are not polynomial-time learnable on average with respect to a target distribution samplable by a c'-sparse $\mathbb{F}_2$-polynomial and a samplable distribution on example distributions with (c, c')-FPT samples.*

2. *There exists a collection of PPRGs in $O(1)$-Fourier-sparse functions iff c-Fourier sparse functions are not polynomial-time learnable on average with respect to a target distribution samplable by a c'-Fourier sparse functions and a samplable distribution on example distributions with (c, c')-FPT samples.*

The generators above still have good parallelism in the sense that each output bit is computable by a constant number of parallel and simple computations (i.e., logical AND or logical XOR).

On obtaining a single PPRG. The theorems above hold only in the case of a collection of PPRGs, and the learning-theoretic characterization of a single PPRG is currently open. Although a collection of PPRGs is standard in parallel cryptography, a single parallel PPRG is still a natural and desirable primitive because it does not require the additional public random strings.

As a third result, we show that if we allow NC^0 circuits to have one top-most XOR-gate with unbounded fan-in, where the other types of gates (i.e., NOT, OR, and AND) have bounded fan-in (we denote this class[4] by $\oplus$-NC^0), then a single PPRG in $\oplus$-NC^0 is characterized by the hardness of learning constant-degree $\mathbb{F}_2$-polynomials on the *uniform* example distribution.

Theorem 6.2.3 (informal)**.** *For any polynomial $r(n)$, the following are equivalent:*

1. *There exists a PPRG in $\oplus$-NC^0.*

2. *Degree-d $\mathbb{F}_2$-polynomials are not polynomial-time learnable on average with respect to a uniform example distribution and a target distribution samplable by a degree-d' $\mathbb{F}_2$-polynomial using $r(n)$-bit random seeds with (d, d')-FPT samples.*

[4]It is not hard to verify that $\oplus$-NC^0 is indeed equivalent to $\mathsf{NC}^0[\oplus]$ (i.e., a class of NC^0 circuits with XOR-gates with unbounded fan-in) and a class of constant-degree $\mathbb{F}_2$-polynomials.

We remark several points. First, in the theorem above, the length of the seeds for selecting a target function is also fixed to some polynomial $r(n)$ independent of the parameters (i.e., degree of $\mathbb{F}_2$-polynomials). This restriction is essential for the result because if we remove this restriction, then unlearnability with FPT samples holds unconditionally even for time-unbounded learners (see Section 6.7.1). Second, the average-case hardness of learning on the uniform example distribution is equivalent to weak pseudorandom functions (WPRFs), where a WPRF is an efficiently samplable family of functions indistinguishable from a random function on inputs passively selected uniformly at random [NR99]. Thus, Theorem 6.2.3 can also be regarded as the equivalence between PPRG and WPRF within the class $\oplus\mathsf{NC}^0$.

Finally, we show that if we consider a general case of samplable distributions of example distributions (instead of the uniform example distribution), then the dichotomy in Theorem 6.2.3 is extended to a collection of PPRGs in $\oplus\mathsf{NC}^0$. In other words, we can characterize the difference between a single PPRG and a collection of PPRGs in $\oplus\mathsf{NC}^0$ by the difference in the generality of example distributions on the hardness of learning.

Theorem 6.2.4 (informal)**.** *For any polynomial $r(n)$, the following are equivalent:*

1. *There exists a collection of PPRGs in $\oplus\mathsf{NC}^0$.*
2. *Degree-d $\mathbb{F}_2$-polynomials are not polynomial-time learnable on average with respect to a target distribution samplable by a degree-d' $\mathbb{F}_2$-polynomial using $r(n)$-bit random seeds and a samplable distribution on example distributions with (d, d')-FPT samples.*

Note that Theorems 6.2.2–6.2.4 also hold in super-polynomial regimes with polynomial security loss.

Theorems 6.2.1–6.2.4 indicate that by selecting a parameterized example distribution and a parameterized target distribution arbitrarily and by assuming the hardness of learning with FPT samples, we can construct a secure parallel PPRG. Conversely, if we believe in PPRGs in the correspondence class, then such a hard-to-learn parameterized setting must exist. However, we remark that Theorems 6.2.1–6.2.4 are general results on the dichotomy between the hardness of learning and parallelly computable PPRGs, and they do not explicitly specify the distributions with respect to which learning is hard on average with FPT samples.

Here, we propose a natural learning task, learning random parity decision trees, whose hardness does not contradict our current knowledge.

Definition 6.2.5 (Learning random parity decision trees)**.** *Let $\mathcal{D} = \{\mathcal{D}_n\}_{n\in\mathbb{N}}$ be an arbitrary example distribution, where $\mathcal{D}_n$ is a distribution on $\{0,1\}^n$ for each $n \in \mathbb{N}$. For any $d \in \mathbb{N}$ and $m\colon \mathbb{N} \to \mathbb{N}$, we define a problem of learning random depth-d parity decision trees (d-LRPDT) on $\mathcal{D}$ with $m(n)$ samples as follows:*

> *Input: samples $\{(x^{(i)}, T(x^{(i)}))\}_{i\in\{1,\dots,m(n)\}}$ and a challenge x_c, where $x^{(1)}, \dots, x^{(m(n))}, x_c \in \{0,1\}^n$ are selected according to $\mathcal{D}_n$, and T is a random parity decision tree of depth d and size 2^d in which each query at internal nodes is $\oplus_{i\in S} x_i$ for a uniformly random subset $S \subseteq \{1,\dots,n\}$ (selected independently for each node) and each leaf is labeled by a uniformly random value in $\{0,1\}$ (selected independently for each leaf).*
> *Output: $T(x_c)$*

For any polynomial $m(n)$ and $p(n)$, we say that d-LRPDT is $(m(n), 1/p(n))$-hard on $\mathcal{D}$ if no randomized polynomial-time algorithm solves d-LRPDT on $\mathcal{D}$ with $m(n)$ samples with probability

at least $1/2 + 1/p(n)$*, i.e., for any randomized polynomial-time algorithm* A *and sufficiently large* $n \in \mathbb{N}$*,*

$$\Pr_{A,\mathcal{D}_n,T}\left[A\left((x^{(1)}, T(x^{(1)})), \ldots, (x^{(m(n))}, T(x^{(m(n))})), x_c\right) = T(x_c)\right] < \frac{1}{2} + \frac{1}{p(n)}.$$

By Theorem 6.2.1, if d-LRPDT is hard with FPT samples on some parametrized sparse example distribution, then a collection of PPRGs exists in NC^0. By inspecting our proof, we show that the sample complexity can be made as small as $n^{1+\epsilon}$ for an arbitrarily small constant $\epsilon > 0$.

Corollary 6.2.6. *Let* $\epsilon \in (0,1)$ *be an arbitrary constant. Suppose that there exist* $d \in \mathbb{N}$ *and an example distribution* $\mathcal{D}$ *such that* d*-LRPDT is hard on* $\mathcal{D}$ *with* $n^{1+\epsilon}$ *samples*[5]*. Then, we can construct parallel PPRGs according to the complexity of* $\mathcal{D}$ *as follows:*

- *If* $\mathcal{D}$ *is* $O(1)$*-sparse and samplable, then a collection of PPRGs in* NC^0 *exists.*
- *If* $\mathcal{D}$ *is the uniform distribution, then a PPRG in* $\oplus$-NC^0 *exists.*
- *If* $\mathcal{D}$ *is samplable, then a collection of PPRGs in* $\oplus$-NC^0 *exists.*

The first and third items hold even for samplable distributions on example distributions.

For instance, as a natural candidate for $O(1)$-sparse example distributions, we propose the uniform distribution over binary strings of Hamming weight $c \in \mathbb{N}$.

Corollary 6.2.7. *If there exist* $c, d \in \mathbb{N}$ *and* $\epsilon \in (0,1)$ *such that* d*-LRPDT is hard on the uniform example distribution over binary strings of Hamming weight* c *with* $n^{1+\epsilon}$ *samples, then a collection of PPRGs in* NC^0 *exists.*

We remark that it is consistent with our knowledge that d-LRPDT cannot be solved. Depth-d parity decision trees are exactly learnable by the Goldreich–Levin algorithm when additional query access to the target function (i.e., membership query) is available [GL89; KM93]. However, it is a central open question whether the membership query is necessary, and d-LRPDT is a natural test case for this question. An efficient learner for random log-depth decision trees was developed by Jackson and Servedio [JS05], but it is unclear whether this algorithm can be extended to the case of random parity decision trees. From Corollary 6.2.6, we propose further learning-theoretic and cryptographic analysis of the hardness of learning parity decision trees as a future research direction. Particularly, one important property of the PPRGs constructed in Corollary 6.2.6 is that the output bits are computed by various predicates. Therefore, they seem to resist an attack that depends on a specific property of one fixed predicate, even in the setting in Corollary 6.2.7.

6.3 Related Work

Applebaum, Barak, and Wigderson [ABW10] proved the hardness of learning $O(\log n)$-junta functions under the existence of PRGs in NC^0 with an additional assumption that (roughly speaking) a small subset of output bits can be embedded indistinguishably with good local expansion. Applebaum and Raykov [AR16] proved the hardness of learning depth-3 AC^0 circuits under the assumption related to polynomial-stretch Goldreich's PRGs, which matches the unconditional upper

[5] For the requirement for the advantage of learning, see Section 6.8.

bound presented in [LMN93]. We remark that their assumption is reducible to a more reliable assumption on Goldreich's OWFs due to the search-to-decision reduction developed in [App13; AR16], where they essentially use the structures of Goldreich's OWFs. Daniely and Vardi [DV21] showed the hardness of learning $\omega(1)$-term DNF formulas and related classes on a product example distribution by assuming Goldreich's PRG for arbitrary polynomial stretch. We remark that our results are incomparable with these previous studies. We assume the existence of the more general cryptographic primitive (i.e., a collection of PPRGs in NC^0) to show the hardness of learning other simple and central classes. This generalization weakens the hardness result to a more general class of example distributions instead of product distributions compared with [DV21], while we can also obtain the opposite direction from the hardness of learning to cryptography. The result of [OST22] on natural properties also differs in the learning setting, particularly natural properties essentially correspond to learning with membership queries on the uniform example distribution [CIKK16].

Blum, Furst, Kearns, and Lipton [BFKL93] constructed OWFs, PRGs, and private-key encryption schemes based on the average-case hardness of learning. To construct PPRGs by using their technique, we need to assume a stronger hardness assumption on learning with membership queries. The use of membership queries was removed by Naor and Reingold [NR99], and we apply the same technique to show one direction in Theorem 6.2.3. Note that the complexity of these PPRGs depends on the complexity of evaluating concept classes. Thus, this approach does not seem to yield a PPRG in NC^0 because if a concept class has the evaluation performed in NC^0, then such a class is trivially learnable. The followup studies [NR06; OS17; San20] discussed relationships between cryptography and hardness of learning in P and $\mathsf{P/poly}$. Other studies [e.g., Reg09] developed various cryptographic schemes based on the hardness of learning linear functions with noise, but it is not clear whether PPRGs in NC^0 are obtained as a consequence of these studies. LRPDT is regarded as a related problem in which we learn parity with noise determined by a constant number of other parities, and it is indeed reducible to learning parity with noise in the case of a uniform example distribution [FGKP09].

With regard to parallel cryptography, the constructions of PRGs in NC^0 were presented by Applebaum, Ishai, and Kushilevitz [AIK06] (sublinear-stretch) and Applebaum, Ishai, and Kushilevitz [AIK08] (linear-stretch). Recently, Ren and Santhanam [RS21] and Liu and Pass [LP21c] characterized the existence of OWF in NC^0 based on the average-case meta-complexity notion, which only yields sublinear-stretch PRGs in NC^0, and PPRGs in NC^0 seem out of reach at present. Some candidates for a collection of PPRGs in NC^0 were studied by Cook, Etesami, Miller, and Trevisan [CEMT14], Bogdanov and Qiao [BQ12], Applebaum, Bogdanov, and Rosen [ABR16], Applebaum [App13], O'Donnell and Witmer [OW14], Applebaum and Lovett [AL18], and Couteau, Dupin, Méaux, Rossi, and Rotella [CDMRR18] based on the framework of Goldreich's OWF [Gol11a]. This type of generator is natural but somewhat restrictive in the sense that all output bits are computed by the same predicate fixed in advance. One advantage of the previous framework is that the security of the generator can be based on a hardness notion of one-wayness, which is more reliable than pseudorandomness [App13].[6] By contrast, an advantage of the framework proposed in this study is that the output bits of the resulting generator are computed by various predicates; thus, it seems to resist an attack that depends on a specific property of one fixed predicate.

We will introduce a key notion of FPT dualization with the junta-sparse condition in Section 6.4,

[6] In terms of learning, the difference between one-wayness and pseudorandomness is similar to the difference between proper learning and improper learning. In general, proper learning is often harder than improper learning, as discussed in Chapter 2.6.

and it seems conceptually related to the analysis of Boolean functions on Hamming balls and slices (i.e., substrings of fixed Hamming weight). Particularly, Filmus and Ihringer [FI19] and Filmus [Fil22] proved that every constant-degree polynomial on a slice is also $O(1)$-junta on the same slice. By contrast, our result can also be rephrased as that every sparse polynomial on a Hamming ball is a *dual* of $O(1)$-junta.

6.4 Techniques

In this section, we present an overview of key notions and proof sketches of the main results.

6.4.1 Proof Techniques for Theorems 6.2.1 and 6.2.2

The key notion to show Theorems 6.2.1 and 6.2.2 is the dualization of concept classes, which was explicitly discussed independently by Applebaum, Barak, and Wigderson [ABW10] and Vadhan [Vad17] and applied (implicitly or explicitly) in recent studies on the hardness of learning [DS16; Dan16; DV21]. Informally, the dualization of a concept class $\mathscr{C}$ consists of two mappings from examples to target functions in $\mathscr{C}$ and from target functions in $\mathscr{C}$ to examples satisfying the following condition. If an example x (resp. a target function $f \in \mathscr{C}$) is mapped to a target function $x^* \in \mathscr{C}$ (resp. an example f^*) by these mappings, then the value of $x^*(f^*)$ is equal to $f(x)$. We refer to this x^* (resp. f^*) as a dual of x (resp. f) and use the superscript $*$ to represent duals. The dualization of a concept class directly enables us to switch the roles of inputs and functions, which was at the core of duality between learning and cryptography, as mentioned in Section 5.2.1. In this context, we can regard the technique in Section 5.2 as an application of the dualization of $\mathsf{P/poly}$ by the universal circuit.

First, we observe that the dualization of a concept class $\mathscr{C}$ provides a relationship between a collection of PRGs and learnability for $\mathscr{C}$. On the one hand, if there exists a collection G of PRGs in $\mathscr{C}$, then we can construct a sample set of size m from the pseudorandom string $y = G(x)$ of length m (where x is a random seed) as $\{(G_i^*, y_i)\}_{i\in[m]}$, where $G_i \in \mathscr{C}$ represents the function computing the i-th bit of G, and G_i^* is its dual. Notice that $x^*(G_i^*) = G_i(x) = y_i$ for each $i \in [m]$. Therefore, if we consider this x^* as a target function for learning $\mathscr{C}$ and the uniform distribution over the samples as the example distribution, any feasible learner cannot distinguish these labels from random labels unless the learner looks at almost all samples in the set. On the other hand, we can obtain a collection of PRGs from the problem of learning $\mathscr{C}$ by translating a sample set $\{(x^{(i)}, f(x^{(i)}))\}_{i\in[m]}$ (where f is a target function) into a generator $G(f^*) = (x^{(1)})^*(f^*) \circ \cdots \circ (x^{(m)})^*(f^*)$. By the equivalence between pseudorandomness and unpredictability [Yao82], if learning $\mathscr{C}$ is hard even with non-negligible advantage, then the value of $G(f^*) = f(x^{(1)}) \circ \cdots \circ f(x^{(m)})$ must be pseudorandom. If we assume that the target distribution is samplable in a complexity class $\mathscr{C}'$ and regard the seed to the sampler as a random seed to G, then we can implement this G in $\mathscr{C} \circ \mathscr{C}'$.

At a high level, we will use the argument above to show Theorems 6.2.1 and 6.2.2. However, there are several obstacles. First, the argument from PRG to the hardness of learning only yields hardness of learning with a fixed sample complexity depending on the stretch of the PRG. Second, more importantly, NC^0 cannot be dualized. Intuitively, for an NC^0-computable $f\colon \{0,1\}^n \to \{0,1\}$ (i.e., f depends on only $O(1)$ coordinates) and input $x \in \{0,1\}^n$, the value of $f(x)$ depends on $\Omega(\log n)$-bit information of f, such as relevant coordinates. Thus, we cannot regard $f(x)$ as a function depending on only $O(1)$-bit information in a representation of f. In Section 6.9, we formally

show the impossibility of the dualization of NC^0 based on the lower bound on communication complexity. Below we present how we deal with these two obstacles.

FPT Dualization

We deal with the first obstacle by assuming polynomial-stretch PRGs. The merit of a PPRG is that we can amplify the stretch of a PRG to an arbitrary polynomial within NC^0 by applying the original generator constant times based on the GGM construction [GGM86]. After applying the original generator computable by a depth-d circuit c times, the depth of the generator increases up to cd, whereas c affects the exponent of the stretch of the PRG. Intuitively, this observation leads to the hardness of learning with FPT samples for a parameter involved in the depth.

To apply the dualization technique above in the parameterized setting, we extend the notion of dualization to the parameterized setting as follows. For any parameterized concept class $\mathscr{C}$, we use a subscript and superscript to refer to an input size and a parameter, respectively.

Definition 6.4.1 (FPT dualizable)**.** *Let $\mathscr{C}^k$ be a parameterized concept class. We say that $\mathscr{C}$ is fixed-parameter tractably (FPT) dualizable if there exist a polynomial $p_{dual}\colon \mathbb{N} \to \mathbb{N}$, computable functions $f_1, f_2\colon \mathbb{N} \to \mathbb{N}$, and polynomial-time computable mappings $g\colon \mathbb{N} \times \{0,1\}^* \to \mathscr{C}$ and $h\colon \mathbb{N} \times \mathscr{C} \to \{0,1\}^*$ such that for any $k, n \in \mathbb{N}$, $x \in \{0,1\}^n$, and $f \in \mathscr{C}_n^k$, the following hold: (i) $g(k,x) \in \mathscr{C}^{f_2(k)}_{f_1(k)\cdot p_{dual}(n)}$, (ii) $h(k,f) \in \{0,1\}^{f_1(k)\cdot p_{dual}(n)}$, and (iii) $(g(k,x))(h(k,f)) = f(x)$.*

We use the notation $x^{*(k)}$ or x^* (resp. $f^{*(k)}$ or f^*) to refer to $g(k,x)$ (resp. $h(k,f)$) in the definition above; e.g., the third condition above can be written as $x^*(f^*) = f(x)$ for each f and x.

Junta-Sparse Condition

At a high level, the idea to overcome the second obstacle is applying the dualization of superclasses of NC^0 and focusing on its substructure, i.e., the correspondence between NC^0 and a subset of strings, particularly in our case, sparse strings. To formalize this idea, we introduce a key condition of FPT dualization named the *junta-sparse condition*, which serves as dualization of NC^0 partially in the actual dualization of the superclass. Intuitively, the junta-sparse condition claims that (i) any $O(1)$-junta function is contained in the concept class, and (ii) $O(1)$-junta functions and strings of constant Hamming weight get interchanged by the FPT dualization. The condition is formally stated as follows:

Definition 6.4.2 (Junta-sparse condition)**.** *Let $\mathscr{C}^k$ be an FPT dualizable class. We say that $\mathscr{C}$ satisfies the junta-sparse condition if the following hold:*

1. *There exist computable functions $g, h\colon \mathbb{N} \to \mathbb{N}$ such that for any $k \in \mathbb{N}$ and any k-junta f, it holds that $f \in \mathscr{C}^{g(k)}$ and $wt(f^*) \leq h(k)$.*
2. *There exists a computable function $g\colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ such that for any $c, k \in \mathbb{N}$ and any $x \in \{0,1\}^*$ with $wt(x) \leq c$, it holds that $x^{*(k)}$ is $g(c,k)$-junta.*

Meta-Theorem

The proof of Theorem 6.2.1 consists of the following two parts. As the first step, we prove meta-theorem which shows that if a parameterized concept class $\mathscr{C}$ is FPT dualizable by mappings

computable in NC^0 and it satisfies the junta-sparse condition, then the existence of a collection of PPRGs in NC^0 corresponds to the average-case hardness of learning $\mathscr{C}$ with FPT samples with respect to (a samplable distribution of) sparse example distributions and an NC^0-samplable target distribution[7]. Note that verifying the condition in the meta-theorem (i.e., dualization with the junta-sparse condition) is purely a puzzle-like problem involved in representation for Boolean functions and directly related to neither learning theory nor cryptography (cf. Section 6.6.3). Namely, if you can solve the puzzle for some concept class $\mathscr{C}$, then it automatically implies the equivalence between the existence of a collection of PPRGs in NC^0 and the average-case hardness of learning $\mathscr{C}$ with sparse data based on our meta-theorem. As the second step to show Theorem 6.2.1, we solve this puzzle, i.e., demonstrate that concept classes in Theorem 6.2.1 (i.e., c-sparse $\mathbb{F}_2$-polynomials, c-Fourier-sparse functions, and depth-d $\{\mathrm{OR},\mathrm{Mod}_m\}$-decision trees) are FPT dualizable by NC^0-computable mappings and satisfy the junta-sparse condition.

We show the outline of the proof of the meta-theorem based on the argument mentioned at the beginning of this subsection.

A collection of PPRGs in $\mathsf{NC}^0 \Rightarrow$ *hardness of learning*: Suppose that there exists a collection G of PPRGs. For contradiction, we assume that there exists an efficient learner L for $\mathscr{C}$ that requires only FPT samples. We amplify the stretch of G by the GGM construction [GGM86] within NC^0, let G' be the amplified generator, and construct the sample set S from the duals of G' and a pseudorandom string $y = G'(x)$. Since G' is computable in NC^0, each function computing each bit of G' is $O(1)$-junta. Thus, by the junta-sparse condition, the Hamming weight of each example is bounded above by a constant (depending on the depth of G'). In addition, since the mappings in FPT dualization are computable in NC^0, the target distribution of the dual of the random seed x is NC^0-samplable. Thus, the learning problem on the uniform distribution over the samples in S is a valid setting for L. Let c be the number of applications of G to construct G'. Then, the sample complexity of L increases in the sense of FPT for c, whereas c affects the exponent of the stretch of G'. Therefore, for a sufficiently large $c \in \mathbb{N}$, the learner L cannot read a large fraction of S. Thus, L can predict some bit in $G'(x)$ from other bits, and this contradicts that G is PRG.

Hardness of learning $\Rightarrow$ *a collection of PPRGs in* NC^0: Suppose that learning $\mathscr{C}$ is hard on average with FPT samples. Since the target distribution is NC^0-samplable, each bit of the representation of $\mathscr{C}$ depends on only constant bits of a random seed. By the technical assumption (in footnote 7) on the FPT upper bound on the length of the representation of $\mathscr{C}$, we can assume that the length of the seed for the target distribution is bounded above by some FPT function. Using the hardness assumption for a sample complexity $m(n)$ polynomially larger than the upper bound on the length of the seed, we construct the collection G of PRGs by taking duals of examples. Remember that the input size of G is the length of the seed for the target distribution, and the output size is $m(n)$. Thus, G has polynomial-stretch. In addition, the Hamming weight of the examples is constant except with negligible probability by the hardness assumption. Thus, by the sparse-junta condition, each bit of G is $O(1)$-junta, and G is implemented in NC^0. Technically, when we consider the advantage in learning, this argument only yields a collection of PPRGs with a fixed indistinguishable parameter. We can convert such a collection of weak PPRGs into a collection of standard PPRGs (with a negligible indistinguishable parameter) by applying the technique by Applebaum and Kachlon [AK19].

Theorem 6.2.2 is shown based on the following observation: If a concept class $\mathscr{C}$ is FPT dualiz-

[7]Strictly speaking, we also need a technical assumption that the length of the binary representation for $\mathscr{C}$ is bounded above by some FPT function.

able and closed under the composition (where the junta-sparse condition is no longer needed), the above argument yields the equivalence between a collection of PPRGs in $\mathscr{C}$ and the average-case hardness of learning $\mathscr{C}$ with FPT samples. See Section 6.6.4 for the formal argument.

6.4.2 Proof Techniques for Theorem 6.2.3

Theorem 6.2.3 shows the equivalence between the existence of a (single) PPRG in $\oplus$-NC^0 and the average-case hardness of learning constant-degree $\mathbb{F}_2$-polynomials with FPT samples with respect to a uniform example distribution and a target distribution samplable by a constant-degree $\mathbb{F}_2$-polynomial. In fact, $\oplus$-NC^0 is equivalent to the class of constant-degree $\mathbb{F}_2$-polynomials because (i) any constant-degree $\mathbb{F}_2$-polynomial is implemented by a $\oplus$-NC^0 circuit that first computes monomials in parallel and takes the summation of them by the top-most XOR gate, and (ii) any $\oplus$-NC^0 circuit is implemented by a constant-degree $\mathbb{F}_2$-polynomial by expressing each sub-circuit connected to the top-most XOR-gate as a constant-degree $\mathbb{F}_2$-polynomial (note that the top-most XOR-gate does not increase the degree of the resulting $\mathbb{F}_2$-polynomial). Therefore, we only need to establish the relationship between a PPRG and learnability within the class of constant-degree $\mathbb{F}_2$-polynomials.

Before presenting the idea, we briefly explain why we cannot apply the dualization techniques in Section 6.4.1 directly to show Theorem 6.2.3. In fact, the class of degree-d $\mathbb{F}_2$-polynomials is simply dualizable as follows: for any degree-d $\mathbb{F}_2$-polynomial $f(x) = \sum_{S:|S|\leq d} f_S \prod_{i\in S} x_i$, where f_S represents the coefficient of f on $\prod_{i\in S} x_i$, we regard the coefficients of f as the input and the value of $\prod_{i\in S} x_i$ as a coefficient on the monomial f_S for each subset S, i.e., the dual of x is a degree-1 $\mathbb{F}_2$-polynomial taking the coefficients of f as the input. An issue is that this dualization is no longer FPT in the sense that each n-input degree-d polynomial is converted into a string of length $\sum_{i=1}^{d} \binom{n}{i} = \Theta(n^d)$. If we apply this dualization in the argument in Section 6.4.1, then a parameter affects the exponent of the sample complexity of learners, and this causes several problems: e.g., in the direction from PPRG to the hardness of learning, we cannot prepare a sufficient number of samples using the GGM construction so that the learner cannot read the entire sample set. In addition, the argument in Section 6.4.1 yields only a collection of PPRGs.

An alternative to show the direction from a PPRG to hardness of learning is to construct an $\mathbb{F}_2$-polynomial pseudorandomly. As a preliminary observation, if we select a polynomial f uniformly at random from all n-input $\mathbb{F}_2$-polynomials of degree d, then for $m = \frac{1}{2}\sum_{i=1}^{d} \binom{n}{i}$ inputs $x^{(1)}, \ldots, x^{(m)} \in \{0,1\}^n$ selected uniformly at random, we can show that the distribution of $f(x^{(1)}), \ldots, f(x^{(m)})$ is statistically close to an m-tuple of random bits even when $x^{(1)}, \ldots, x^{(m)}$ are given. In the formal proof, we verify this by applying the results obtained by Ben-Eliezer, Hod, and Lovett [BHL12]. For now, we assume this. Then, we observe that even if we select a degree-d $\mathbb{F}_2$-polynomial f by a pseudorandom string generated by a PPRG, the labels of the sample set $\{(x^{(i)}, f(x^{(i)}))\}$ must be computationally indistinguishable from random labels. By the equivalence of pseudorandomness and unpredictability [Yao82], such a pseudorandom $\mathbb{F}_2$-polynomial f must be unpredictable.

Based on the argument above, we can create a hard learning problem with FPT samples based on a PPRG G, as follows. For contradiction, we assume that there exists an efficient learner L that requires only FPT samples. Then, we use the GGM construction to amplify the stretch of G, let G' denote the amplified PRG, and select a pseudorandom $\mathbb{F}_2$-polynomial using G'. Remember that the number c of applications of G affects the exponent of the stretch of G'. Thus, for each $d \in \mathbb{N}$, we can select a sufficiently large c such that a degree-d pseudorandom $\mathbb{F}_2$-polynomial can be selected by G'. Note that G' is still computable by an $\mathbb{F}_2$-polynomial of degree d^c. We regard

this G' as a sampling algorithm for selecting a target function in degree-d $\mathbb{F}_2$-polynomials. For the degree-d pseudorandom $\mathbb{F}_2$-polynomial, we can retrieve $\frac{1}{2}\sum_{i=1}^{d}\binom{n}{i} = \Theta(n^d)$ samples that are hard to predict. By contrast, each d determines c and the degree of the sampling algorithm for the target distribution; thus, d affects the required number of samples only in the FPT sense. Therefore, by taking a sufficiently large d, we can prepare a sufficient number of samples for L, and L yields an efficient adversary for G' and G. This is a contradiction.

To show the opposite direction from the average-case hardness of learning to a PPRG, we apply the idea presented by Naor and Reingold [NR99]. First, we observe that for each constant-degree $\mathbb{F}_2$-polynomial f and input x, the value of $f(x)$ is evaluated by a constant-degree $\mathbb{F}_2$-polynomial taking x and the binary representation of f as the input (where we naturally assume that each f is represented by the coefficients of f). Then, the construction of a PPRG G is outlined as follows. We use the hardness assumption for a sample complexity $m(n)$ sufficiently larger than $(n+r(n))^2$, where $r(n)$ is the upper bound on the seed length for the target distribution in Theorem 6.2.3. Let $R = n + r(n)$. Then, G selects R^2 examples $x^{(1)}, \ldots, x^{(R^2)}$ and R^2 target functions $f^{(1)}, \ldots, f^{(R^2)}$ according to the hard example distribution and target distribution by using its own random seed. Then, G outputs R^4 bits $f^{(i)}(x^{(j)})$ for each $i, j \in \{1, \ldots, R^2\}$ as a pseudorandom string. We can prove the pseudorandomness of G using the hybrid argument and the equivalence between unpredictability and pseudorandomness [Yao82]. Since G requires only a $R^2(n + r(n))$-bit random seed to select the examples and the target functions, G stretches an R^3-bit random seed into an R^4-bit pseudorandom string. Thus, G has polynomial-stretch. Note that we apply the standard padding technique to obtain a PPRG defined on all input lengths. Since the sampling algorithm for the target distribution and the evaluation algorithm are computable by constant-degree $\mathbb{F}_2$-polynomials, this generator is implemented by a constant-degree $\mathbb{F}_2$-polynomial by taking composition. Thus, we obtain a PPRG computable by a constant-degree $\mathbb{F}_2$-polynomial. Note that the construction in the formal proof is more complicated because we apply the XOR lemma to amplify the success probability of the adversary to the desired advantage of a learner. For details, see Section 6.7.1.

6.4.3 Proof Ideas for Theorem 6.2.4

Theorem 6.2.4 shows the equivalence of a collection of PPRGs in $\oplus\mathsf{NC}^0$ and the average-case hardness of learning constant-degree $\mathbb{F}_2$-polynomials with FPT samples with respect to (a samplable distribution of) example distributions and a target distribution samplable by a constant-degree $\mathbb{F}_2$-polynomial. One direction from the average-case hardness of learning to a collection of PPRGs is shown in the same way as in Section 6.4.2 except that the sampling algorithm for the example distributions is simulated during preprocessing, where the examples are hardwired in the generator.

We present a rough idea to show the other direction from a collection of PPRGs to the hardness of learning. Note that we cannot apply the technique in Section 6.4.2 because the sampler of generators cannot be implemented in constant-degree $\mathbb{F}_2$-polynomials in general, and the sampling algorithm for selecting a pseudorandom $\mathbb{F}_2$-polynomial is not always implemented in constant-degree $\mathbb{F}_2$-polynomials. Thus, we take the strategy based on FPT dualization again. As discussed in Section 6.4.2, it is unclear whether FPT dualization of constant-degree $\mathbb{F}_2$-polynomials is feasible. However, to show the direction from a PPRG to hardness of learning based on the argument in Section 6.4.1, the type of functions we need to dualize is restrictive, i.e., composite functions of the original pseudorandom generator G (in the GGM construction). We apply this observation to avoid the obstacle involved in the dualization of general constant-degree $\mathbb{F}_2$-polynomials.

The outline follows the argument in Section 6.4.1. Let G' be the collection of PPRGs obtained

by applying G c times to amplify the stretch. We create the sample set from G' and a pseudorandom string $y = G'(x)$, where each example corresponds to the dual of the function computing each bit of G'. Intuitively, for each position i, we define the dual of the i-th bit of G' as c concatenated descriptions of G that are relevant for computing the i-th bit of G'. Then, we consider a target function as a constant-degree $\mathbb{F}_2$-polynomial that computes the description of G' by taking the composition of the given descriptions of G and then applies the random seed x, where we regard this x to be hardwired by another constant-degree $\mathbb{F}_2$-polynomial given x as the input. We regard the latter $\mathbb{F}_2$-polynomial as the sampling algorithm for the target distribution. Consequently, we can prevent the dependence of c and the degree d of G' on the exponent of the input size and the sample complexity in learning. By contrast, c affects the exponent of the stretch of G'. Thus, based on the similar argument as in Section 6.4.1, we can show the average-case hardness of learning by selecting sufficiently large c. We will present the details in Section 6.7.2.

6.5 Additional Preliminaries

For any $m \in \mathbb{N}$, we define a symmetric function $\mathrm{Mod}_m \colon \{0,1\}^* \to \{0,1\}$ as $\mathrm{Mod}_m(x) = 1$ iff $x \equiv 0 \bmod m$. For any $n, d \in \mathbb{N}$ with $d \le n$, let $\binom{n}{\le d} = \sum_{i=0}^{d} \binom{n}{i} = O(n^d)$.

We use the following lemma.

Lemma 6.5.1 ([BHL12, Claim 2.4]). *For any $\beta \in (0,1)$, there exists a constant $\gamma \in (0,1)$ such that for any $m, d \in \mathbb{N}$ and for any sufficiently large $n \in \mathbb{N}$, $\binom{m}{\le d} \le \beta \cdot \binom{n}{\le d}$ implies $m \le n(1-\gamma/d)$.*

Let $\mathscr{C} = \{\mathscr{C}_n\}_{n\in\mathbb{N}}$ be an arbitrary class of functions (i.e., a complexity class), where $\mathscr{C}_n \subseteq \{f \colon \{0,1\}^n \to \{0,1\}\}$ for each $n \in \mathbb{N}$. When we discuss the computability in $\mathscr{C}$ in this chapter, we implicitly assume its uniformity, i.e., we say that a family of multi-output functions $f = \{f_n\}_{n\in\mathbb{N}}$, where $f \colon \{0,1\}^n \to \{0,1\}^{m(n)}$, is computable in $\mathscr{C}$ if there exists a polynomial-time algorithm A such that for any $n \in \mathbb{N}$ and $i \in [m(n)]$, the algorithm $A(1^n, i)$ outputs the description of a function $g_i \in \mathscr{C}_n$ such that $g_i(x)$ is the i-th bit of $f_n(x)$ for any input $x \in \{0,1\}^n$. Let NC^0 (resp. $\oplus\text{-}\mathsf{NC}^0$) be the complexity class of constant-depth circuits (resp. constant-depth circuits in which the top-most gate can be a $\oplus$-gate with unbounded fan-in).

6.5.1 Boolean Functions and Representations

In this chapter, we consider a distribution on functions samplable in low complexity. In such cases, the choice of binary encodings of the functions may affect the results because the translation between two different representations may be infeasible in low complexity. Thus, we specify the binary representations for concept classes in a natural manner as follows.

$\mathbb{F}_2$-polynomials

Any Boolean-valued function $f \colon \{0,1\}^n \to \{0,1\}$ has a unique representation as a polynomial in $\mathbb{F}_2$ obtained by expanding $f(x) = \sum_{a\in\mathbb{F}_2^n} f(a)\mathbb{1}(x = a) = \sum_{a\in\mathbb{F}_2^n} f(a) \prod_{i\in[n]}(x_i + a_i + 1)$ under operations of $\mathbb{F}_2$.

For each $S \subseteq [n]$ and $x \in \mathbb{F}_2^n$, let $x^S = \prod_{i\in S} x_i$. For each $\mathbb{F}_2$-polynomial $p \colon \mathbb{F}_2^n \to \mathbb{F}_n$ and $S \subseteq [n]$, we use the notation p_S to refer to the coefficient of p on S, i.e., $p(x) = \sum_S p_S x^S$. We define the degree of an $\mathbb{F}_2$-polynomial p as the maximum number d such that there exists a subset S of coordinates such that $|S| = d$ and $p_S = 1$. Then, we specify the binary representation of degree-d

$\mathbb{F}_2$-polynomials naturally by a string of length $\binom{n}{\leq d}$ concatenating all coefficients on S with $|S| \leq d$ in some canonical order.

The following lemma plays a key role in the proof of Theorem 6.2.3.

Lemma 6.5.2 ([BHL12, Lemma 1.4]). *For any $n, m \in \mathbb{N}$ and any 2^m distinct points $x_1, \ldots, x_{2^m} \in \mathbb{F}_2^n$, the following set is a linear subspace of $\mathbb{F}_2^{2^m}$ and the dimension is at least $\binom{m}{\leq d}$:*

$$\left\{v^p \in \mathbb{F}_2^{2^m} : p \text{ is a degree-}d\ \mathbb{F}_2\text{-polynomial and } v^p_i = p(x_i) \text{ for each } i \in [2^m]\right\}.$$

Fourier Representations

When we consider the Fourier representation of Boolean functions, we regard any Boolean-valued function $f\colon \{0,1\}^n \to \{0,1\}$ as a function mapping from $\{0,1\}^n$ to $\{-1,1\}$ by considering $(-1)^{f(x)}$. For each $\alpha \in \{0,1\}^n$, we define a function $\chi_\alpha\colon \{0,1\}^n \to \{-1,1\}$ as $\chi_\alpha(x) = (-1)^{\langle x,\alpha\rangle}$, where $\langle\cdot,\cdot\rangle$ denotes the inner product in $\mathbb{F}_2^n$. Then, any Boolean function $f\colon \{0,1\}^n \to \{-1,1\}$ has a unique representation of the form $f(x) = \sum_{S\subseteq[n]} \widehat{f}(\alpha)\chi_\alpha(x)$, where $\widehat{f}(\alpha) = \mathbb{E}_{x\sim\{0,1\}^n}[f(x)\chi_\alpha(x)]$ and is called a Fourier coefficient of f on α. For further background on Fourier analysis, refer to the textbook by O'Donnell [ODo14].

For any function $f\colon \{0,1\}^n \to \{-1,1\}$, the Fourier sparsity of f is defined as $|\{S \subseteq [n] : \widehat{f}(S) \neq 0\}|$. For any $s \in \mathbb{N}$ and any function f of Fourier sparsity s, each Fourier coefficient $\widehat{f}(\alpha)$ takes the form of $M_\alpha/2^{\lceil \log s\rceil}$, where $M_\alpha \in \{-2^{\lceil\log s\rceil}, \ldots, 0, \ldots, 2^{\lceil\log s\rceil}\}$ [GOSSW11; ODo14, Exercise 3.32]. Thus, we assume that each n-input function f of Fourier sparsity s is represented by an $O(ns\log s)$-bit string, where each term in f is represented by a tuple of $\lceil\log s\rceil + 1$ bits indicating the coefficient (i.e., M_α above) and n bits indicating the coordinates that are contained in the term (i.e., α above). For instance, $f(x_1, \ldots, x_n) = x_1 \vee x_2$ is 4 Fourier-sparse function and represented in this form as $((-2, 0^n), (2, 10^{n-1}), (2, 010^{n-2}), (2, 110^{n-2}))$.

Decision Trees and Extensions

A decision tree (DT) is a representation of Boolean functions and is defined as a rooted binary tree in which the internal nodes are labeled by a variable x_i, and the leaves are labeled by $\{0,1\}$. For an n-input DT T and input $x \in \{0,1\}^n$, the value of $T(x)$ is determined as follows: T queries the value in x according to the label at the root, and if the answer is true (resp. false), then T looks at the right (resp. left) subtree and repeats the same process for the subtree. T repeats this until it reaches some leaf and then outputs the binary label of the reached leaf. We define the depth of DT as the maximum length of a path from the root to the leaves.

For any (family of) symmetric function f (e.g., OR and Mod_m), we define an f-decision tree (f-DT) in the same manner as above except that each internal node is labeled by the query of the form $f(x_{i_1}, \ldots, x_{i_k})$ for some $k \in [n]$ and $\{i_1, \ldots, i_k\} \subseteq [n]$ (instead of x_i).

Without loss of generality, we can assume that the number of internal nodes of any depth-d f-DT is exactly $2^d - 1$ by adding dummy nodes, where nothing is queried, and a configuration automatically proceeds to the false subtree. We also assume a standard canonical ordering in $2^d - 1$ nodes (root to leaves) and 2^d leaves (left to right). Then, for any (family of) symmetric function f, we naturally specify the binary representation of n-input f-decision trees of depth f as a $(2^d - 1)\cdot n + 2^d$-bit string consisting of $2^d - 1$ strings in $\{0,1\}^n$ that represent the sets of relevant coordinates in $[n]$ for $2^d - 1$ internal nodes and 2^d binary labels on leaves.

6.5.2 Learning Models

We define a parameterized concept class $\mathscr{C} = \{\mathscr{C}^k\}_{k\in\mathbb{N}}$ as a family of concept classes such that $\mathscr{C}^k \subseteq \mathscr{C}^{k+1}$ for each $k \in \mathbb{N}$. Note that we often use a subscript and a superscript to refer to input size and a parameter, respectively.

In this chapter, we discuss the prediction model in the BFKL model [BFKL93] in a more fine-grained way than Chapter 2, which is stated as follows. For convenience, we regard the time bound of a learning algorithm as a function in the input length of a target function (i.e., the example size) instead of a function in the input length of learning algorithms.

Definition 6.5.3 (Prediction in BFKL model)**.** *Let $\mathscr{C}$ be a concept class. Let $\mathcal{D} = \{\mathcal{D}_n\}_{n\in\mathbb{N}}$ and $\mathcal{F} = \{\mathcal{F}_n\}_{n\in\mathbb{N}}$ be families of distributions, where $\mathcal{D}_n$ is a distribution on $\{0,1\}^n$ and $\mathcal{F}_n$ is a distribution on $\mathscr{C}_n$. For any functions $t, m\colon \mathbb{N} \to \mathbb{N}$ and $\gamma\colon \mathbb{N} \to (0, 1/2)$, we say that $\mathscr{C}$ is (t, m, γ)-learnable on average with respect to $\mathcal{D}$ and $\mathcal{F}$ if there exists a randomized algorithm L such that for all sufficiently large $n \in \mathbb{N}$,*

$$\Pr_{L,f,x^{(1)},\ldots,x^{(m(n))},x_c}\left[L((x^{(1)}, f(x^{(1)})), \ldots, (x^{(m(n))}, f(x^{(m(n))})), x_c) \text{ outputs } f(x_c) \text{ in time } t(n)\right] \geq \frac{1}{2}+\gamma(n),$$

where $x^{(1)}, \ldots, x^{(m(n))}, x_c \sim \mathcal{D}_n$ and $f \sim \mathcal{F}_n$.

We refer to $\mathcal{D}$ (resp. F) as an example (resp. a target) distribution. We also refer to f, x_c, and γ above as a target function, a challenge, and an advantage, respectively.

Without loss of generality, we ignore the cases in which $m(n) > t(n)$. Next, we define the key notion of this work, i.e., FPT sample complexity.

Definition 6.5.4 (FPT samples)**.** *For $c \in \mathbb{N}$, let $k_1, \ldots, k_c$ be parameters on a concept class $\mathscr{C}$ and classes of example distributions and target distributions. For any functions $t\colon \mathbb{N} \to \mathbb{N}$ and $\gamma\colon \mathbb{N} \to (0, 1/2)$, we say that $\mathscr{C}$ is (t, γ)-learnable on average with $(k_1, \ldots, k_c)$-FPT samples if there exists a function $m(n, k_1, \ldots, k_c) = f(k_1, \ldots, k_c) \cdot n^{O(1)}$ for some $f\colon \mathbb{N}^c \to \mathbb{N}$ such that for any choice of $k_1, \ldots, k_c \in \mathbb{N}$ and any choice of an example distribution $\mathcal{D}$ and target distribution $\mathcal{F}$ that satisfy the settings of the parameters, $\mathscr{C}$ is $(t, m_{k_1,\ldots,k_c}, \gamma)$-learnable on average with respect to $\mathcal{D}$ and $\mathcal{F}$, where $m_{k_1,\ldots,k_c}(n) := m(n, k_1, \ldots, k_c)$.*

We define distributions on example distributions as example distributions samplable with shared randomness to introduce the average-case variant of distribution specific learning.

Definition 6.5.5 (Samplable with shared randomness)**.** *We say that an example distribution is samplable with shared randomness if there exists a polynomial-time sampling algorithm S such that for any example size $n \in \mathbb{N}$, examples in $\{0,1\}^n$ are selected identically and independently according to $S(U_{\mathsf{poly}(n)}; r)$, where r is an auxiliary random string selected uniformly at random from $\{0,1\}^{\mathsf{poly}(n)}$ at the initiation and shared through sampling processes.*

Note that learning on example distribution samplable with shared randomness is the notion sandwiched between distribution-free learning and distribution-specific learning in the following sense. Any distribution-free learner that succeeds on all (unknown) example distributions also succeeds on any example distribution samplable with shared randomness regardless of the choice of shared randomness. In addition, if there exists a learner that succeeds on D for each example distribution D samplable with shared randomness, then there exists a distribution-specific learner that succeeds on D' for each samplable example distribution D'.

We also discuss RRHS-refutation in the BFKL model.

Definition 6.5.6 (RRHS-refutation in BFKL model)**.** *Let $\mathscr{C}$ be a concept class, $\mathcal{D}$ be an example distribution, and $\mathcal{F}$ be a target distribution on $\mathscr{C}$. For functions $t, m\colon \{0,1\}^n \to \mathbb{N}$ and $\gamma\colon \mathbb{N} \to (0,1/2)$, we say that $\mathscr{C}$ is (t,m,γ)-random-right-hand-side-refutable (RRHS-refutable) on average with respect to $\mathcal{D}$ and $\mathcal{F}$ if there exists a randomized $t(n)$-time algorithm A such that for any $n \in \mathbb{N}$,*

$$\Pr_{A,x,f}\left[A((x^{(1)}, f(x^{(1)})), \dots, (x^{(m(n))}, f(x^{(m(n))}))) = 1\right] - \Pr_{A,x,b}\left[A((x^{(1)}, b^{(1)}), \dots, (x^{(m(n))}, b^{(m(n))})) = 1\right] \geq \gamma(n),$$

where $f \sim \mathcal{F}_n$, $x^{(i)} \sim \mathcal{D}_n$, and $b^{(i)} \sim \{0,1\}$ for each $i \in [m(n)]$.

We use one direction from the hardness of learning to the hardness of RRHS-refuting, which follows from Yao's next-bit generator [Yao82].

Theorem 6.5.7 ([Yao82; Vad17])**.** *Let $m\colon \mathbb{N} \to \mathbb{N}$ and $\gamma\colon \mathbb{N} \to (0,1/2)$ be any polynomial-time computable functions. Let $\mathcal{D}$ be an arbitrary example distribution and $\mathcal{F}$ be an arbitrary target distribution on a concept class $\mathscr{C}$. Then, there exists a polynomial q such that for any time-bound function $t(n)$, if $\mathscr{C}$ is not $(t(n), m(n), \gamma(n))$-learnable on average with respect to $\mathcal{D}$ and $\mathcal{F}$, then $\mathscr{C}$ is not $(t(n)/q(n), m(n), m(n)\gamma(n))$-RRHS-refutable on average with respect to $\mathcal{D}$ and $\mathcal{F}$.*

We introduce the following useful fact, which follows from the XOR lemma (cf. [Nan21a; GNW11, Sections 2 and 3]).

Fact 6.5.8. *For any polynomial $m^{\oplus}, p$, there exist polynomials m, ℓ, q and a randomized algorithm* **Boost** *such that for any example distribution $\mathcal{D}_{ex}$ and any samplable target distribution $\mathcal{D}_{targ}$ on a concept class $\mathscr{C}$, the following hold.*

- *ℓ is determined by only p (i.e., independent of $m^{\oplus}$).*
- **Boost** *is given $m(n)$ samples and a challenge according to $\mathcal{D}_{ex}$ and $\mathcal{D}_{targ}$ with a description of a randomized algorithm $L^{\oplus}$ and outputs a prediction for the challenge.*
- *We define a concept class $\mathscr{C}^{\oplus}$ by $\mathscr{C}^{\oplus}_n = \mathcal{C}^{\oplus}_{n'\ell(n')}$, where n' is the maximum integer satisfying $n'\ell(n') \leq n$ and*

$$\mathcal{C}^{\oplus}_{n\ell(n)} = \left\{ f(x^{(1)} \circ \dots \circ x^{(\ell(n))}) := \bigoplus_{i,j \in [\ell(n)]} f^{(i)}(x^{(j)}) \middle| f^{(i)} \in \mathscr{C}_n, x^{(j)} \in \{0,1\}^n \right\}.$$

Let $\mathcal{D}^{\oplus}_{ex}$ and $\mathcal{D}^{\oplus}_{targ}$ be families of distributions, where $(\mathcal{D}^{\oplus}_{ex})_n$ and $(\mathcal{D}^{\oplus}_{targ})_n$ are the distributions of $x^{\oplus} := x^{(1)} \circ \dots \circ x^{(\ell(n'))}$ and $f^{\oplus}(x^{\oplus}) := \bigoplus_{i,j} f^{(i)}(x^{(j)})$ for $x^{(1)}, \dots, x^{(\ell(n'))} \sim (\mathcal{D}_{ex})_{n'}$ and $f^{(1)}, \dots, f^{(\ell(n'))} \sim (\mathcal{D}_{targ})_{n'}$, respectively (where n' is the maximum integer satisfying $n'\ell(n') \leq n$). If $L^{\oplus}$ $(t(n), m^{\oplus}(n), 1/p^{\oplus}(n))$-learns $\mathscr{C}^{\oplus}$ on average with respect $\mathcal{D}^{\oplus}_{ex}$ and $\mathcal{D}^{\oplus}_{targ}$ for some polynomial $p^{\oplus}$, then **Boost** *$(t(n\ell(n))q(n), m(n), 1/2 - 1/p(n))$-learns $\mathscr{C}$ on average with respect to $\mathcal{D}_{ex}$ and $\mathcal{D}_{targ}$ for any sufficiently large n.*

6.5.3 Pseudorandom Generator

Remember that, for convenience, we regard the time-bound of adversaries as a function in the input length of a generator (i.e., the length of the hidden random seed) instead of a function in the input length of adversaries.

Definition 6.5.9 (Polynomial-stretch PRG and weak PRG)**.** *Let $t\colon \mathbb{N} \to \mathbb{N}$ be any time-bound function. We say that a family $G = \{G_n\}_{n\in\mathbb{N}}$, where $G_n\colon \{0,1\}^n \to \{0,1\}^{\ell(n)}$ for some function $\ell\colon \mathbb{N} \to \mathbb{N}$, is an (infinitely often) pseudorandom generator (PRG) against $t(n)$-time adversaries if $\ell(n) > n$ and for any randomized $t(n)$-time algorithm A, there exist infinitely many $n \in \mathbb{N}$ such that*

$$\left| \Pr_{A,U_n}[A(1^n, G_n(U_n)) = 1] - \Pr_{A,U_{\ell(n)}}[A(1^n, U_{\ell(n)}) = 1] \right| \leq \mathsf{negl}(n).$$

In addition, we say that a PRG G is a polynomial-stretch PRG (PPRG) if $\ell(n) > n^{1+\epsilon}$ holds for some constant $\epsilon > 0$.

For any polynomial $p(n)$, we say that G is a weak PRG with indistinguishable parameter $1/p(n)$ against $t(n)$-time adversaries if $\ell(n) > n$ and for any randomized $t(n)$-time algorithm A, there exist infinitely many $n \in \mathbb{N}$ such that

$$\left|\Pr[A(1^n, G_n(U_n)) = 1] - \Pr[A(1^n, U_{\ell(n)}) = 1]\right| \leq 1/p(n).$$

We usually omit the subscript n from the notation above. Instead, we use the notation G_i for $i \in [n]$ to refer to the function computing the i-th bit of G. We also often omit 1^n from the input to adversaries.

Note that any generator in NC^0 has a constant locality because any depth-d circuit only depends on at most 2^d coordinates of the input.

We also extend the definition above to a collection of pseudorandom generators.

Definition 6.5.10 (A collection of PRGs)**.** *We say that a family $G = \{G_{n,z}\}_{n\in\mathbb{N},z\in\{0,1\}^{\mathsf{poly}(n)}}$, where $G_{n,z}\colon \{0,1\}^n \to \{0,1\}^{\ell(n)}$ for some function $\ell\colon \mathbb{N} \to \mathbb{N}$, is a collection of PRGs against $t(n)$-time adversaries if (i) $\ell(n) > n$, (ii) for any (n, z), the binary representation[8] of $G_{n,z}$ is computable from $(1^n, z)$ in time $\mathsf{poly}(n)$, and (iii) for any randomized $t(n)$-time algorithm A, there exist infinitely many $n \in \mathbb{N}$ such that*

$$\left| \Pr_{z\sim\{0,1\}^{\mathsf{poly}(n)},A,U_n}[A(G_{n,z}, G_{n,z}(U_n)) = 1] - \Pr_{z\sim\{0,1\}^{\mathsf{poly}(n)},A,U_{\ell(n)}}[A(G_{n,z}, U_{\ell(n)}) = 1] \right| \leq \mathsf{negl}(n),$$

where the input $G_{n,z}$ refers to the binary representation of $G_{n,z}$. Moreover, if $\ell(n) > n^{1+\epsilon}$ holds for some constant $\epsilon > 0$, then we say that G is a collection of PPRGs.

We also define a collection of weak PRGs in the same manner as Definition 6.5.9.

We often omit the subscripts n and z from the notation above and refer to a choice of z as a choice of G.

We introduce two useful theorems shown in earlier studies. The first theorem shows the way to amplify the stretch of PPRG by applying the original PPRG repeatedly constant time.

[8] Specifically, when we discuss a collection of PPRGs in a class $\mathcal{C}$ of Boolean functions, this is the binary representation for $\mathcal{C}$.

Theorem 6.5.11 ([GGM86]). *For any function $G = \{G_n\}_{n\in\mathbb{N}}$, where $G\colon \{0,1\}^n \to \{0,1\}^{n^{1+\epsilon}}$ for some constant $\epsilon > 0$, and for any constants $c \in [\lceil \epsilon^{-1} \rceil]$ and $d \in \mathbb{N}$, we define functions $G^{(0,c)} = \{G_n^{(0,c)}\}_{n\in\mathbb{N}}$ and $G^{(d)} = \{G_n^{(d)}\}_{n\in\mathbb{N}}$, where $G^{(0,c)}\colon \{0,1\}^n \to \{0,1\}^{n^{1+c\epsilon}}$ and $G^{(d)}\colon \{0,1\}^n \to \{0,1\}^{n^{d+1}}$, inductively as follows:*

$$
\begin{aligned}
G^{(0,1)}(x) &= G(x)\\
G_n^{(0,c)}(x) &= G_n(G_n^{(0,c-1)}(x)_{[1:n]}) \circ G_n(G_n^{(0,c-1)}(x)_{[n+1:2n]}) \circ \cdots \circ G_n(G_n^{(0,c-1)}(x)_{[n^{1+(c-1)\epsilon}-n+1:n^{1+(c-1)\epsilon}]})\\
G^{(1)}(x) &= G_n^{(0,\lceil \epsilon^{-1} \rceil)}(x)_{[1:n^2]}\\
G_n^{(d)}(x) &= G_n^{(1)}(G_n^{(d-1)}(x)_{[1:n]}) \circ G_n^{(1)}(G_n^{(d-1)}(x)_{[n+1:2n]}) \circ \cdots \circ G_n^{(1)}(G_n^{(d-1)}(x)_{[n^d-n+1:n^d]}).
\end{aligned}
$$

For each $d \in \mathbb{N}$, there exists a polynomial q such that for any time-bound function t, if G is a PPRG against $t(n)$-time adversaries, then $G^{(d)}$ is also a PPRG against $t(n)/q(n)$-time adversaries. Furthermore, this also holds for a collection of PRGs.

The second theorem shows that any weak PPRG with indistinguishable parameter $n^{-\Theta(1)}$ can be converted into a PPRG (with negligible indistinguishable parameter) without loss of constant locality.

Theorem 6.5.12 ([AK19]). *For any constant $d \in \mathbb{N}$, $a > 0$, and $\epsilon, \epsilon' > 0$, there exist $d' \in \mathbb{N}$, a polynomial q, and $\delta \in (0,1)$ such that any collection of weak PPRGs of stretch $n^{1+\epsilon}$ and indistinguishable parameter $1/n^a$ computable in depth-d NC^0 against $t(n)$-time adversaries can be converted into a collection of PPRGs of stretch $n^{1+\epsilon'}$ in depth-d' NC^0 against $t(n^\delta)/q(n)$-time adversaries.*

6.6 Learning vs. PPRGs in Constant-Parallel Time

In this section, we show Theorems 6.2.1 and 6.2.2, which are formally stated as follows.

Theorem 6.6.1. *For any $a > 1$, the following are equivalent:*

1. *There exists a collection of PPRGs in NC^0.*
2. *c-sparse $\mathbb{F}_2$-polynomials are not polynomial-time learnable on average with advantage n^{-a} with respect to a c'-sparse example distribution samplable with shared randomness a target distribution samplable by a depth-d NC^0 circuit with (c, c', d)-FPT samples.*
3. *c-Fourier-sparse functions are not polynomial-time learnable on average with advantage n^{-a} with respect to a c'-sparse example distribution samplable with shared randomness and a target distribution samplable by a depth-d NC^0 circuit with (c, c', d)-FPT samples.*
4. *For any $f \in \{\mathrm{OR}\} \cup \{\mathrm{MOD}_m : m \in \mathbb{N} \setminus \{1\}\}$, depth-$d$ f-decision trees are not polynomial-time learnable on average with advantage n^{-a} with respect to a c-sparse example distribution samplable with shared randomness and a target distribution samplable by a depth-d' NC^0 circuit with (d, c, d')-FPT samples.*

Theorem 6.6.2. *For any $a > 1$, the following hold:*

1. *There exists a collection of PPRGs in $O(1)$-sparse $\mathbb{F}_2$-polynomials iff c-sparse $\mathbb{F}_2$-polynomials are not polynomial-time learnable on average with advantage n^{-a} with respect to an example distribution samplable with shared randomness and a target distribution samplable by a c'-sparse $\mathbb{F}_2$-polynomial with (c, c')-FPT samples.*

2. *There exists a collection of PPRGs in $O(1)$-Fourier-sparse functions iff c-Fourier sparse functions are not polynomial-time learnable on average with advantage n^{-a} with respect to an example distribution samplable with shared randomness and a target distribution samplable by a c'-Fourier sparse functions with (c, c')-FPT samples.*

6.6.1 FPT Dualization and Junta-Sparse Condition

First, we review the key notions for showing Theorem 6.6.1.

Definition (FPT dualizable)**.** *Let $\mathscr{C}^k$ be a parameterized concept class. We say that $\mathscr{C}$ is FPT dualizable if there exist a polynomial $p_{dual}\colon \mathbb{N} \to \mathbb{N}$, computable functions $f_1, f_2\colon \mathbb{N} \to \mathbb{N}$, and polynomial-time computable mappings $g\colon \mathbb{N} \times \{0,1\}^* \to \mathscr{C}$ and $h\colon \mathbb{N} \times \mathscr{C} \to \{0,1\}^*$ such that for any $k, n \in \mathbb{N}$, $x \in \{0,1\}^n$, and $f \in \mathscr{C}_n^k$, the following hold: (i) $g(k,x) \in \mathscr{C}^{f_2(k)}_{f_1(k)\cdot p_{dual}(n)}$, (ii) $h(k,f) \in \{0,1\}^{f_1(k)\cdot p_{dual}(n)}$, and (iii) $(g(k,x))(h(k,f)) = f(x)$.*

Moreover, for parameterized classes $\mathscr{C}^k$ and $\mathscr{D}^\ell$, we say that $\mathscr{C}$ is FPT dualizable in $\mathscr{D}$ if (i) $\mathscr{C}$ is FPT dualizable and (ii) there exists a computable function $l : \mathbb{N} \to \mathbb{N}$ such that for any $k \in \mathbb{N}$, it holds that $g(k,\cdot)$ and $h(k,\cdot)$ are computable in $\mathscr{D}^{l(k)}$.

We use the notation $x^{*(k)}$ or x^* (resp. $f^{*(k)}$ or f^*) to refer to $g(k,x)$ (resp. $h(k,f)$) in the definition above. For example, the third condition above can be rewritten as $x^*(f^*) = f(x)$ for any $f \in \mathscr{C}$ and $x \in \{0,1\}^*$.

Definition (Junta-sparse condition)**.** *Let $\mathscr{C}^k$ be an FPT dualizable class. We say that $\mathscr{C}$ satisfies the junta-sparse condition if the following hold:*

1. *There exist computable functions $g, h\colon \mathbb{N} \to \mathbb{N}$ such that for any $k \in \mathbb{N}$ and any k-junta f, it holds that $f \in \mathscr{C}^{g(k)}$ and $wt(f^*) \le h(k)$.*

2. *There exists a computable function $g\colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ such that for any $c, k \in \mathbb{N}$ and any $x \in \{0,1\}^*$ with $wt(x) \le c$, it holds that $x^{*(k)}$ is $g(c,k)$-junta.*

6.6.2 Meta-Theorems

We present the meta-theorems for Theorem 6.6.1.

Theorem 6.6.3 (PPRG in $\mathsf{NC}^0 \Rightarrow$ hardness of learning)**.** *Let $p(n)$ be an arbitrary polynomial and $\mathscr{C}^k$ be a parameterized class that is FPT dualizable in NC^0 (parameterized by depth) and satisfies the junta-sparse condition. There exist a polynomial $q(n)$ and a constant $\epsilon > 0$ such that for any time-bound function $t(n)$, if there exists a collection of PPRGs in NC^0 against $t(n)$-time adversaries, then $\mathscr{C}$ is not $(t(n^\epsilon)/q(n), 1/p(n))$-learnable on average with respect to a c-sparse example distribution samplable with shared randomness and a target distribution samplable by a depth-d NC^0 circuit with (k, c, d)-FPT samples.*

Proof. Let G be a collection of PPRGs with locality d_0, and let $\mathcal{G}$ be its generator, i.e., $\mathcal{G}(1^n; r)$ outputs a description of G in polynomial time for a random seed $r \in \{0,1\}^{\mathsf{poly}(n)}$. Let f_1, f_2, p_{dual} be the functions in Definition 6.4.1 for the FPT dualization of $\mathscr{C}$. Then, we select a constant $\epsilon \in (0,1]$ such that $(\log n \cdot p_{dual}(n))^\epsilon \leq n$, i.e., $\log n \cdot p_{dual}(n) \leq n^{1/\epsilon}$.

We fix an FPT sample-complexity function $m_{k,c,d}(n) = f_m(k,c,d) \cdot p_m(n)$ arbitrarily, where $f_m \colon \mathbb{N} \times \mathbb{N} \times \mathbb{N} \to \mathbb{N}$, and p_m is a polynomial. Then, we select a sufficiently large $D \in \mathbb{N}$ such that $n^D \geq p(\log n \cdot p_{dual}(n)) \cdot p_m(\log n \cdot p_{dual}(n))$. We construct a collection of PPRGs $G^{(D)}$ in Theorem 6.5.11 based on G. It is easily verified that the locality of $G^{(D)}$ is at most some constant D', i.e., each output of $G^{(D)}$ is computable by a D'-junta function. By the junta-sparse condition, any D'-junta function is contained in $\mathscr{C}^k$ for some $k \in \mathbb{N}$. The description of $G_i^{(D)}$ for each $i \in [n^{D+1}]$ as a function in $\mathscr{C}^k$ is computable in polynomial time by using $\mathcal{G}$.

Now, we introduce the hard problem for learning $\mathscr{C}$. We specify the example distribution $\mathcal{D}_{ex}$ by the following sampling algorithm S using shared randomness. On input 1^n and shared randomness $r \in \{0,1\}^{\mathsf{poly}(n)}$, the sampling algorithm S generates the description of G by executing $\mathcal{G}(1^n; r)$. Then, S selects $i \sim [n^{D+1}]$ by an (unshared) random seed, and computes $f \in \mathscr{C}^k$ corresponding to the D'-junta function $G_i^{(D)}$. Finally, S outputs the dual $f^* \in \{0,1\}^{f_1(k) \cdot p_{dual}(n)}$ of f as an example. We also define the target distribution $\mathcal{D}_{targ}$ as the distribution of $x^* \in \mathscr{C}^{f_2(k)}_{f_1(k) \cdot p_{dual}(n)}$ for randomly selected $x \sim \{0,1\}^n$.

By the junta-sparse condition, $\mathcal{D}_{ex}$ is c-sparse for some constant $c \in \mathbb{N}$. Since $\mathscr{C}$ is FPT dualizable in NC^0, the target distribution $\mathcal{D}_{targ}$ is samplable by a depth-d NC^0 circuit for some constant $d \in \mathbb{N}$. Therefore, if we assume that $\mathscr{C}$ is $(t(n^\epsilon)/q(n), 1/p(n))$-learnable on average with sample complexity $m_{k,c,d}$ for contradiction, there exists an algorithm that succeeds in $(t(n^\epsilon)/q(n), m_{k,c,d}(n), 1/p(n))$-learning $\mathscr{C}^k$ on average with respect to $\mathcal{D}_{ex}$ and $\mathcal{D}_{targ}$.

By selecting sufficiently large $q(n)$, we will show that for any time-bound function T, any learner L that $(T(n), m_{k,c,d}(n), 1/p(n))$-learns $\mathscr{C}^k$ (on average with respect to $\mathcal{D}_{ex}$ and $\mathcal{D}_{targ}$) can be converted into a $T(n^{1/\epsilon}) \cdot q(n^{1/\epsilon})$-time adversary that breaks G. Since $(t((n^{1/\epsilon})^\epsilon)/q(n^{1/\epsilon})) \cdot q(n^{1/\epsilon}) = t(n)$, any algorithm that succeeds in $(t(n^\epsilon)/q(n), m_{k,c,d}(n), 1/p(n))$-learning $\mathscr{C}^k$ yields a $t(n)$-time adversary G. This contradicts that G is a PRG against $t(n)$-time adversaries.

First, we construct an adversary A for $G^{(D)}$ as follows: On input $w \in \{0,1\}^{n^{D+1}}$ and the description of $G^{(D)}$ (note that w is a pseudorandom string generated by $G^{(D)}$ or a truly random string), A simulates the example distribution $\mathcal{D}_{ex}$ by selecting a random index $i \sim [n^{D+1}]$, computing the D'-junta function corresponding to $G_i^{(D)}$ and its dual (for simplicity, we let G_i^* denote this dual string of length $N := f_1(k) \cdot p_{dual}(n)$), and generating a sample (G_i^*, w_i). After generating $m_{k,c,d}(N)$ samples, A also generates a challenge $G_{i_c}^*$ for $i_c \sim [n^{D+1}]$ and feeds it to L. If L outputs some prediction $b \in \{0,1\}$, then A checks whether $b = w_{i_c}$. If so, A outputs 1; otherwise, it outputs 0. We remark that the running time of A is bounded above by $\mathsf{poly}(n) \cdot T(N)$.

In the case in which $w \sim G^{(D)}(x)$ for $x \sim \{0,1\}^n$, we have $w_i = G_i^{(D)}(x) = x^*(G_i^*)$ for all i. Therefore, the simulated samples are valid for the target function x^*. Furthermore, it is not hard to verify that A executes L on the example distribution $\mathcal{D}_{ex}$ and the target distribution $\mathcal{D}_{targ}$. Therefore, we have

$$\Pr_{A,U_n,G}[A(G^{(D)}, G^{(D)}(U_n)) = 1] = \Pr_{L,\mathcal{D}_{ex},\mathcal{D}_{targ}}[L \text{ succeeds in learning}] \geq \frac{1}{2} + \frac{1}{p(N)}.$$

By contrast, in the case in which $w \sim \{0,1\}^{n^{D+1}}$, the labels in the simulated samples are selected truly at random. Because any learning algorithm cannot guess a random label not contained in the

given samples better than a random guess, i.e., with success probability $1/2$, we have

$$\begin{aligned}\Pr_{A,U_n,G}[A(G^{(D)}, U_{n^{D+1}})) = 1] &= \Pr_{L,\mathcal{D}_{ex}}[L \text{ succeeds in learning}] \\ &\leq \frac{1}{2} \cdot \left(1 - \frac{m_{k,c,d}(N)}{n^{D+1}}\right) + 1 \cdot \frac{m_{k,c,d}(N)}{n^{D+1}} \\ &= \frac{1}{2} + \frac{m_{k,c,d}(N)}{2n^{D+1}} \\ &\leq \frac{1}{2} + \frac{f_m(k,c,d) \cdot p_m(N)}{2n \cdot p(\log n \cdot p_{dual}(n)) \cdot p_m(\log n \cdot p_{dual}(n))}\end{aligned}$$

Therefore, for sufficiently large n,

$$\begin{aligned}\Pr_{A,U_n,G}[A(G^{(D)}, U_{n^{D+1}})) = 1] &\leq \frac{1}{2} + \frac{f_m(k,c,d) \cdot p_m(N)}{2n \cdot p(N) \cdot p_m(N)} \\ &\leq \frac{1}{2} + \frac{1}{2p(N)}\end{aligned}$$

and the advantage of A is at least

$$\left(\frac{1}{2} + \frac{1}{p(N)}\right) - \left(\frac{1}{2} + \frac{1}{2p(N)}\right) \geq \frac{1}{2p(N)} \geq \frac{1}{2p(n \cdot p_{dual}(n))}.$$

Thus, A successfully breaks $G^{(D)}$.

By Theorem 6.5.11, the adversary A for $G^{(D)}$ can be converted into an adversary A' for G such that the running time of A is bounded above by $\mathsf{poly}(n) \cdot T(N) \leq q(n^{1/\epsilon}) \cdot T(\log n \cdot p_{dual}(n)) \leq q(n^{1/\epsilon}) \cdot T(n^{1/\epsilon})$ for a sufficiently large polynomial q. □

Next, we prove the opposite direction.

Theorem 6.6.4 (hardness of learning ⇒ PPRG in NC^0)**.** *Let $p(n) = n^{\Theta(1)}$ be a polynomial, and let $\mathscr{C}^k$ be a parameterized class that is FPT dualizable in NC^0 (parameterized by depth). Assume that for any $k \in \mathbb{N}$ and sufficiently large $n \in \mathbb{N}$, the length of the representation for $\mathscr{C}^k$ is at most $p(n)^{1-\epsilon}$ for some constant $\epsilon \in (0,1)$. Then, there exist a polynomial $q(n)$ and a constant $\delta > 0$ such that for any time-bound function $t(n)$, if $\mathscr{C}$ is not $(t(n), 1/p(n))$-learnable on average with respect to a c-sparse example distribution samplable with shared randomness and a target distribution samplable by a depth-d NC^0 circuit with (k,c,d)-FPT samples, then there exists a collection of PPRGs in NC^0 against $t(n^\delta)/q(n)$-time adversaries.*

Proof. We use the hardness assumption for the sample complexity function $m_{k,c,d}(n) = m(n) := p(n)^{1-\epsilon/2}$ (i.e., independent of parameters). Then, there exist constants $k, c, d \in \mathbb{N}$, an example distribution $\mathcal{D}_{ex}$, and a target distribution $\mathcal{D}_{targ}$ for the hard problem of learning $\mathscr{C}^k$, where $\mathcal{D}_{ex}$ is c-sparse and samplable with shared randomness, and $\mathcal{D}_{targ}$ is samplable by a depth-d NC^0 circuit. We remark that the length of the representation for $\mathscr{C}$ is at most $p(n)^{1-\epsilon}$. Since $\mathcal{D}_{targ}$ is samplable by a depth-d NC^0 circuit, each bit of such a representation is determined by a constant number of random seeds for $\mathcal{D}_{targ}$. Therefore, without loss of generality, we assume that the length of random bits for $\mathcal{D}_{targ}$ is at most $\ell(n) = p(n)^{1-3\epsilon/4} (= n^{\Theta(1)})$ for sufficiently large $n \in \mathbb{N}$.

We construct a collection of weak PPRGs in NC^0, where the indistinguishable parameter is $p(\ell^{-1}(n))^{-\epsilon/2} = n^{-\Theta(1)}$. Then, we apply Theorem 6.5.12 to obtain a collection of (standard) PPRGs in NC^0.

By Theorem 6.5.7, the hardness assumption implies that $\mathscr{C}^k$ is not RRHS-refutable on average with respect to $\mathcal{D}_{ex}$ and $\mathcal{D}_{targ}$ with $m(n)$ samples and advantage $p(n)^{-\epsilon/2}$. Now, we introduce the generator $\mathcal{G}$ of PPRGs. On input $1^{\ell(n)}$, the generator $\mathcal{G}$ first generates $m(n)$ examples $x^{(1)}, \dots, x^{(m(n))} \sim \mathcal{D}_{ex}$. Since $\mathcal{D}_{ex}$ is samplable with shared randomness, $\mathcal{G}$ can perfectly simulate $\mathcal{D}_{ex}$ in polynomial time. Then, $\mathcal{G}$ computes their duals $(x^{(1)})^*, \dots, (x^{(m(n))})^*$, where the input to each $(x^{(m(n))})^*$ is the dual of the target function selected according to $\mathcal{D}_{targ}$. By the junta-sparse condition, these duals $(x^{(1)})^*, \dots, (x^{(m(n))})^*$ are $O(1)$-junta except with negligible probability when the dual of a target function is given as input. Since $\mathcal{D}_{targ}$ is samplable by a depth-d NC^0 circuit whose input is the random seed $r \in \{0,1\}^{\ell(n)}$, and the dual of the target function is computable in NC^0, by considering the composition of $(x^{(1)})^*, \dots, (x^{(m(n))})^*$, the NC^0 circuit computing the dual, and the NC^0 circuit sampling the target function, we make $m(n)$ NC^0 circuits $G_1(r), \dots, G_{m(n)}(r)$, where each G_i corresponds to $(x^{(i)})^*$. Finally, $\mathcal{G}$ outputs $G(r) := G_1(r) \circ \cdots \circ G_{m(n)}(r)$ as the description of the NC^0-computable generator.

We remark that the above-mentioned generator is only defined for input size $\ell(n)$. This can be converted into a generator defined for all input sizes n by the standard technique, i.e., for a given n-bit random seed, the generator uses only $\ell(n')$ bits, where n' is the maximum integer such that $\ell(n') \leq n$ (for details, refer to the textbook by Goldreich [Gol01]). Let G denote the generator. Then, the length N of the output of G is at least

$$N = m(n') = p(n')^{1-\epsilon/2} = \ell(n')^{1+\frac{\epsilon}{4-3\epsilon}} > \ell(n'+1)^{1+\epsilon'} > n^{1+\epsilon'}$$

for some $\epsilon' \in (0, \frac{\epsilon}{4-3\epsilon})$ and any sufficiently large n. Thus, G has polynomial-stretch.

Next, we show that G satisfies the security condition of a weak pseudorandom generator by contradiction. We assume that there exists a $T(n)$-time adversary A such that

$$\left| \Pr_{\mathcal{G},A,U_n}[A(G, G(U_n)) = 1] - \Pr_{\mathcal{G},A,U_N}[A(G, U_N) = 1] \right| > 1/p(\ell^{-1}(n))^{\epsilon/2}. \tag{6.1}$$

Now, we construct a refuting algorithm R for $\mathscr{C}^k$ as follows. On input $(x^{(1)}, b^{(1)}), \dots, (x^{(m(n))}, b^{(m(n))})$, the algorithm R constructs the generator G in the same way as $\mathcal{G}$, i.e., R computes $(x^{(i)})^*$ and the composed function G_i for each $i \in [m(n)]$. Then, R executes $A(G, b^{(1)} \circ \cdots \circ b^{(m(n))})$ and returns the same answer. We remark that each $x^{(i)}$ is selected according to $\mathcal{D}_{ex}$. Thus, R correctly simulates the distribution of the generator G. In the case in which $f \sim \mathcal{D}_{targ}$ and $b^{(i)} = f(x^{(i)})$ for each i, we have $b^{(i)} = f(x^{(i)}) = (x^{(i)})^*(f^*) = G_i(r)$, where r is the seed for selecting f, and the distribution of $b^{(1)} \circ \cdots \circ b^{(m(n))}$ corresponds to $G(U_{\ell(n)})$. By contrast, in the case in which $b^{(i)} \sim \{0,1\}$ for each i, the distribution of $b^{(1)} \circ \cdots \circ b^{(m(n))}$ corresponds to a uniform distribution. Therefore, by (6.1), R refutes $\mathscr{C}^k$ on $\mathcal{D}_{ex}$ and $\mathcal{D}_{targ}$ with $m(n)$ samples and advantage grater than $1/p(\ell^{-1}(\ell(n)))^{\epsilon/2} = p(n)^{-\epsilon/2}$.

By Theorem 6.5.7, the refuter R can be converted to a learner with advantage $1/p(n)$. By selecting a sufficiently large polynomial $q(n)$ and a sufficiently large constant $a > 1$, the running time of the learner is bounded above by $q(n) \cdot T(\ell(n)) \leq q(n) \cdot T(n^a)$. Thus, by letting $\delta = 1/a$, any $t(n^\delta)/q(n)$-time adversary for G is converted into a learning algorithm that works in time $q(n) \cdot t(n^{\delta \cdot a})/q(n^a) \leq t(n)$ with advantage $1/p(n)$, which is a contradiction. □

6.6.3 FPT Dualizable Classes with Junta-Sparse Condition

In this section, we present FPT dualization in NC^0 with the junta-sparse condition for c-sparse $\mathbb{F}_2$-polynomials, c-Fourier-sparse functions, and depth-d $\{\mathrm{OR}, \mathrm{Mod}_m\}$-decision trees. Then, we can

show Theorem 6.6.1 by applying Theorems 6.6.3 and 6.6.4 for all polynomial time-bounds $t(n)$. To apply Theorem 6.6.4, we leverage the fact that for any $a > 0$ and the parameter of the class, the length of the binary representations of target functions is at most n^{1+a} for sufficiently large input size $n \in \mathbb{N}$.

c-Sparse $\mathbb{F}_2$-Polynomials

For each c-sparse $\mathbb{F}_2$-polynomial $f = M_1 + \cdots + M_c$, where each M_i represents a monomial, and for each input $x \in \{0,1\}^n$, we define their duals as a binary string $f^* \in \{0,1\}^{cn+c}$ and a $2c$-sparse $\mathbb{F}_2$-polynomial x^*. For simplicity, we assume that f^* is indexed by $\{0, \cdots, n\} \times [c]$ instead of $[cn+c]$. Then, f^* and x^* is determined as follows.

$$f^*_{(i,j)} = \begin{cases} \mathbb{1}(x_i \in M_j) & \text{if } i \in [n] \\ \mathbb{1}(M_j \equiv 1) & \text{if } i = 0 \end{cases}$$
$$x^*(f^*) = \sum_{j \in [c]} \prod_{i : x_i = 1} f^*_{(i,j)} + \sum_{j \in [c]} f^*_{(0,j)}.$$

The dualization above is trivially computable in NC^0. The correctness is verified as follows:

$$\begin{aligned} x^*(f^*) &= \sum_{j \in [c]} \prod_{i : x_i = 1} f^*_{(i,j)} + \sum_{j \in [c]} f^*_{(0,j)} \\ &= \sum_{j \in [c]} \left(\prod_{i : x_i = 1} \mathbb{1}(x_i \in M_j) + \mathbb{1}(M_j \equiv 1) \right) \\ &= \sum_{j \in [c]} M_j(x) \\ &= f(x). \end{aligned}$$

In addition, the junta-sparse condition is verified as follows:

Lemma 6.6.5. *c-sparse $\mathbb{F}_2$-polynomials satisfy the junta-sparse condition by the FPT dualization in NC^0 above.*

Proof. (1.) Any n-input k-junta function is represented as an n-input $\mathbb{F}_2$-polynomial of degree k and sparsity 2^k. It is not hard to verify that for any degree-k 2^k-sparse $\mathbb{F}_2$-polynomial, the Hamming weight of its dual f^* is at most $2^k \cdot k$.

(2.) For any $n, c, c' \in \mathbb{N}$ and $x \in \{0,1\}^n$ with $wt(x) \leq c'$, the dual x^{*c} depends on only $c \cdot wt(x) + c \leq cc' + c$ coordinates. □

c-Fourier-Sparse Functions

For each $x \in \{0,1\}^n$ and each c-Fourier-sparse function $f = M_1 2^{-\lceil \log c \rceil} \chi_{\alpha_1} + \cdots + M_c 2^{-\lceil \log c \rceil} \chi_{\alpha_c}$, where $M_i \in \{-2^{\lceil \log c \rceil}, \ldots, 2^{\lceil \log c \rceil}\}$ and $\alpha_i \in \{0,1\}^n$ for each $i \in [c]$, we define their duals as a binary string $f^* \in \{0,1\}^{(\lceil \log c \rceil + n + 1)c}$ and a function x^* of Fourier sparsity $c' := 2c\lceil \log c \rceil$.

For each $i \in [c]$, let $b^i \in \{0,1\}^{\lceil \log c \rceil}$ be the binary representation of the absolute value of M_i. Then, f^* consists of c triples of b^i, α_i, and $b^i_{neg} \in \{0,1\}$, where $b^i_{neg} = 1$ iff $M_i < 0$.

We also specify x^* as $x^* = \sum_{(i,j)\in[c]\times[\lceil \log c\rceil]} N_{i,j}2^{-\lceil \log c'\rceil}\chi_{i,j} + N'_{i,j}2^{-\lceil \log c'\rceil}\chi'_{i,j}$, where $N_{i,j}, N'_{i,j} \in \{-2^{\lceil \log c'\rceil}, \dots, 2^{\lceil \log c'\rceil}\}$ and $\chi'_{i,j}, \chi'_{i,j} \colon \{0,1\}^{(\lceil \log c\rceil n+1)c} \to \{-1,1\}$ are determined as follows:

$$\begin{aligned}
N_{i,j} &= 2^{\lceil \log c'\rceil + j - 1 - \lceil \log c\rceil} \ (\leq 2^{\lceil \log c'\rceil - 1}) \\
N'_{i,j} &= -N_{i,j} = -2^{\lceil \log c'\rceil + j - 1 - \lceil \log c\rceil} \\
\chi_{i,j}(f^*) &= (-1)^{b^i_{neg} + \sum_{k:x_k=1}(\alpha_i)_k} \\
\chi'_{i,j}(f^*) &= (-1)^{b^i_j}\chi_{i,j}(f^*) = (-1)^{b^i_j + b^i_{neg} + \sum_{k:x_k=1}(\alpha_i)_k}.
\end{aligned}$$

It is not hard to verify that the dualization above is computable in NC^0. The correctness is verified as follows:

$$\begin{aligned}
x^*(f^*) &= \sum_{(i,j)\in[c]\times[\lceil \log c\rceil]} N_{i,j}2^{-\lceil \log c'\rceil}\chi_{i,j}(f^*) + N'_{i,j}2^{-\lceil \log c'\rceil}\chi'_{i,j}(f^*) \\
&= \sum_{i\in[c]}\sum_{j\in[\lceil \log c\rceil]} 2^{j-1-\lceil \log c\rceil}(-1)^{b^i_{neg}+\sum_{k:x_k=1}(\alpha_i)_k} - 2^{j-1-\lceil \log c\rceil}(-1)^{b^i_j + b^i_{neg}+\sum_{k:x_k=1}(\alpha_i)_k} \\
&= \sum_{i\in[c]}(-1)^{\sum_{k:x_k=1}(\alpha_i)_k}2^{-\lceil \log c\rceil}(-1)^{b^i_{neg}} \sum_{j\in[\lceil \log c\rceil]} 2^j \cdot (1-(-1)^{b^i_j})/2 \\
&= \sum_{i\in[c]}(-1)^{\langle x,\alpha_i\rangle}2^{-\lceil \log c\rceil} \cdot (-1)^{b^i_{neg}} \sum_{j\in[\lceil \log c\rceil]} 2^j \cdot b^i_j \\
&= \sum_{i\in[c]} \chi_{\alpha_i}(x) \cdot 2^{-\lceil \log c\rceil} M_i \\
&= f(x).
\end{aligned}$$

In addition, the junta-sparse condition is verified as follows:

Lemma 6.6.6. *c-Fourier-sparse functions satisfy the junta-sparse condition by the FPT dualization in NC^0 above.*

Proof. (1.) Based on the unique Fourier representation, any n-input k-junta function is represented as a degree-k function of Fourier sparsity at most 2^k. It is not hard to verify that for any degree-k 2^k-Fourier sparse function, the Hamming weight of its dual f^* is at most $2^k \cdot (\lceil \log 2^k\rceil + k + 1) = 2^k \cdot (2k+1)$.

(2.) For any $n, c, c' \in \mathbb{N}$ and $x \in \{0,1\}^n$ with $wt(x) \leq c'$, the dual x^{*c} depends on only $2c\lceil \log c\rceil \cdot (2 + wt(x)) \leq 2c\lceil \log c\rceil(2+c')$ coordinates. □

Depth-d Mod_m-Decision Trees and OR-Decision Trees

In this subsection, we present the FPT dualization for Mod_m-DT that satisfies the junta-sparse condition. Note that the case of OR-DT follows in the same way.

For each $x \in \{0,1\}^n$ and depth-d Mod_m-DT T, we define their duals as a binary string $T^* \in \{0,1\}^{(2^d-1)n+2^d}$ and a depth-$(d+1)$ Mod_m-DT x^*. For simplicity, we assume that T^* consists of a tuple $t \in \{0,1\}^{(2^d-1)n}$ and $\ell \in \{0,1\}^{2^d}$, and t is indexed by $[2^d-1]\times[n]$ instead of $[(2^d-1)n]$. Let $\{j_1, \dots, j_c\} = \{j \in [n] : x_j = 1\}$, where $c := wt(x)$. Then, T^* (i.e., t and ℓ) and x^* are defined as

follows:

$$t_{i,j} = \mathbb{1}(x_j \text{ is relevant to the query at node } i)$$
$$\ell_i = (\text{the label at leaf } i),$$

and for any $i \in [2^{d+1}-1]$ and $j \in [2^{d+1}]$,

$$(\text{the query at node } i \text{ in } x^*) = \begin{cases} \mathrm{Mod}_{\mathrm{m}}(t_{i,j_1}, \dots, t_{i,j_c}) & i \le 2^d - 1 \\ \mathrm{Mod}_{\mathrm{m}}(\ell_{i-(2^d-1)}) & i \ge 2^d \end{cases}$$
$$(\text{the label at leaf } j \text{ in } x^*) = \mathbb{1}(\text{leaf } j \text{ is the false subtree of its parent node}).$$

The dualization above is computable in NC^0. We also verify the correctness. On evaluating $x^*(T^*)$, any answer to the query at node $i \in [2^d - 1]$ is consistent with the answer to the query at node i in $T(x)$ because

$$\mathrm{Mod}_{\mathrm{m}}(t_{i,j_1}, \dots, t_{i,j_c}) = \mathrm{Mod}_{\mathrm{m}}(x_1 \wedge t_{i,1}, \dots, x_n \wedge t_{i,n}) = \mathrm{Mod}_{\mathrm{m}}(x_{k_1^i}, \dots, x_{k_\cdot^i}),$$

where

$$\{k_1^i, \dots, k_\cdot^i\} = \{k \in [n] : x_k \text{ is relevant to the query at node } i \text{ in } T\}.$$

For any $i \in [2^{d+1}-1] \setminus [2^d-1]$, the answer to the query at node i is $\mathrm{Mod}_{\mathrm{m}}(\ell_{i-(2^d-1)}) = \neg\ell_{i-(2^d-1)}$ for any $m \ge 2$. Note that x^* outputs 1 (i.e., true) iff the answer to the query at depth $d+1$ is false. Thus, $x^*(T^*)$ is consistent with $T(x)$.

Furthermore, the junta-sparse condition is verified as follows:

Lemma 6.6.7. *For any $m \ge 2$, depth-d Mod_m-DT satisfies the junta-sparse condition by the FPT dualization in NC^0 above.*

Proof. (1.) Any n-input k-junta function is represented as a depth-k Mod_m-DT, where each query is represented as $\mathrm{Mod}_m(x_i) = \neg x_i$ for some $i \in [n]$. It is not hard to verify that the Hamming weight of the dual of such a Mod_m-DT is at most $(2^d - 1) + 2^d$.

(2.) For any $n, d, c \in \mathbb{N}$ and $x \in \{0,1\}^n$ with $wt(x) \le c$, the dual x^{*d} depends on only $(2^d - 1) \cdot wt(x) + 2^d \le (2^d - 1)c + 2^d$ coordinates. □

6.6.4 Relaxed Hardness Assumption

In this section, we present the meta-theorem for Theorem 6.6.2. First, we introduce a natural condition of parameterized concept classes.

Definition 6.6.8 (junta-composition condition)**.** *Let $\mathscr{C}^k$ be a parameterized class. We say that $\mathscr{C}$ satisfies the junta-composition condition if the following hold:*

1. *For any $k, n', n \in \mathbb{N}$ with $n' \le n$, it holds that $\mathscr{C}^k_{n'} \subseteq \mathscr{C}^k_n$ (i.e., paddable with dummy inputs).*
2. *There exists a computable $g\colon \mathbb{N} \to \mathbb{N}$ such that any k-junta function is contained in $\mathscr{C}^{g(k)}$ for each $k \in \mathbb{N}$.*
3. *There exists a computable $g\colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ such that for any $k, k', n, n' \in \mathbb{N}$, $f^{(1)}, \dots, f^{(n)} \in \mathscr{C}^k_{n'}$, and $f' \in \mathscr{C}^{k'}_n$, the composite function $f''\colon \{0,1\}^{n'} \to \{0,1\}$ defined as $f''(x) = f'(f^{(1)}(x), \dots, f^{(n)}(x))$ is contained in $\mathscr{C}^{g(k,k')}$. In addition, the representation of f'' is computable from $f^{(1)}, \dots, f^{(n)}$, and f' in polynomial time.*

It is easily verified that c-sparse $\mathbb{F}_2$-polynomials and c-Fourier-sparse functions satisfy the junta-composition condition.

Suppose that G is a weak PPRG of output length $n^{1+\epsilon}$, where each bit is computable in $\mathscr{C}^k$ satisfying the junta-composition condition. For the translation to a (standard) PPRG of output length $n^{1+\epsilon'}$ in Theorem 6.5.12, we only need the following operations (for details, refer to [App13; AK19]): Let $f^{(1)}, \dots, f^{(n^{1+\epsilon})}$ be the functions computing each bit of G. Then, the operations are either of

- $f^{(i)}(x) := f^{(i_0)}(f^{(i_1)}(x), \dots, f^{(i_n)}(x))$ for some $i_0 \in [n^{1+\epsilon}]$ and $i_1, \dots, i_n < i$ (for amplifying the stretch);
- $f^{(i)}(x^{(1)}, \dots, x^{(t)}) := f^{(i_1)}(x^{(1)}) \oplus \cdots \oplus f^{(i_t)}(x^{(t)})$ for some $t \in \mathbb{N}$ and $i_1, \dots, i_t < i$ (for amplifying the unpredictability); or
- $f^{(i)}(x^{(1)}, \dots, x^{(\mathsf{poly}(n))}, r) := g(f^{(i_1)}(x^{(1)}), \dots, f^{(i_{\mathsf{poly}(n)})}(x^{(\mathsf{poly}(n))}), r)$ for some $i_1, \dots, i_{\mathsf{poly}(n)} < i$, $r \in \{0,1\}^{\mathsf{poly}(n)}$, and some $O(1)$-junta function g (for applying the extractor presented in [AK19]),

and the resulting PPRG is computable by $f^{(i_1)}, \dots, f^{(i_{n^{1+\epsilon'}})}$ for some indices $i_1, \dots, i_{n^{1+\epsilon'}}$.

If the $f^{(1)}, \dots, f^{(n^{1+\epsilon})} \in \mathscr{C}^k$ and $\mathscr{C}^k$ satisfies the junta-composition condition, it is not hard to verify that each $f^{(i)}$ is contained in $\mathscr{C}^{k'}$ for some k' by induction. Therefore, we have the following analog of Theorem 6.5.12.

Theorem 6.6.9 ([AK19])**.** *Let $\mathscr{C}^k$ be a parameterized class satisfying the junta-composition condition. For any $k \in \mathbb{N}$, $a > 0$, and $\epsilon, \epsilon' > 0$, there exist $k' \in \mathbb{N}$ (computable from k), a polynomial q, and $\delta \in (0,1)$ such that any collection of weak PPRGs in $\mathscr{C}^k$ of stretch $n^{1+\epsilon}$ and indistinguishable parameter $1/n^a$ against $t(n)$-time adversaries can be converted into a collection of PPRGs in $\mathscr{C}^{k'}$ of stretch $n^{1+\epsilon'}$ against $t(n^\delta)/q(n)$-time adversaries.*

Now, we present the meta-theorem for Theorem 6.6.2, where we only assume the FPT dualization and the junta-composition condition.

Theorem 6.6.10 (PPRG in $\mathscr{C} \Rightarrow$ hardness of learning $\mathscr{C}$)**.** *Let $p(n)$ be an arbitrary polynomial, and let $\mathscr{C}^k$ be a parameterized class that is FPT dualizable in a parameterized class $\mathscr{F}^\ell$ and satisfies the junta-composition condition. There exist a polynomial $q(n)$ and a constant $\epsilon > 0$ such that for any time-bound function $t(n)$, if there exist $k \in \mathbb{N}$ and a collection of PPRGs in $\mathscr{C}^k$ against $t(n)$-time adversaries, then $\mathscr{C}^k$ is not $(t(n^\epsilon)/q(n), 1/p(n))$-learnable on average with respect to an example distribution samplable with shared randomness and an $\mathscr{F}^\ell$-samplable target distribution with (k,ℓ)-FPT samples.*

Proof. (sketch.) The theorem follows in the same way as Theorem 6.6.3. The proof is outlined as follows:

We assume the existence of a collection of PPRGs in $\mathscr{C}^k$ for some $k \in \mathbb{N}$. Then, for each FPT sample complexity m, we amplify the stretch sufficiently by applying Theorem 6.6.9 so that any learning algorithm with sample complexity m cannot read all the output bits of the generator (where we use the junta-composition condition to apply Theorem 6.6.9). We remark that each bit of the generator is computable in $\mathscr{C}^{k'}$ for some k'. Next, we specify the hard learning problem where the example distribution $\mathcal{D}_{ex}$ is the uniform distribution over the duals of the functions computing

the generator, and the target distribution $\mathcal{D}_{targ}$ is the distribution of $x^{*k'}$ for $x \sim \{0,1\}^n$. It is not hard to verify that $\mathcal{D}_{ex}$ is samplable with shared randomness. By FPT dualization in $\mathscr{F}^\ell$, the target distribution $\mathcal{D}_{targ}$ is over $\mathscr{C}^{k''}$ for some $k'' \in \mathbb{N}$ and $\mathscr{F}^\ell$-samplable for some $\ell \in \mathbb{N}$. Therefore, this is a valid case for learning, and any learner succeeds in learning on average with respect to $\mathcal{D}_{ex}$ and $\mathcal{D}_{targ}$ with sample complexity m can be converted into the adversary for the collection of PPRGs, as in the proof of Theorem 6.6.3. This is a contradiction. □

Next, we show the opposite direction.

Theorem 6.6.11 (hardness of learning $\mathscr{C} \Rightarrow$ PPRG in $\mathscr{C}$)**.** *Let $p(n) = n^{\Theta(1)}$ be a polynomial and $\mathscr{C}^k$ be a parameterized class that is FPT dualizable in $\mathscr{C}^k$ and satisfies the junta-composition condition. Assume that for any $k \in \mathbb{N}$ and sufficiently large $n \in \mathbb{N}$, the length of the representation for $\mathscr{C}^k$ is at most $p(n)^{1-\epsilon}$ for some constant $\epsilon \in (0,1)$. Then, there exists a polynomial $q(n)$ and a constant $\delta > 0$ such that for any time-bound function $t(n)$, if $\mathscr{C}^k$ is not $(t(n), 1/p(n))$-learnable on average with respect to an example distribution samplable with shared randomness and a $\mathscr{C}^{k'}$-samplable target distribution with (k,k')-FPT samples, then there exists a collection of PPRGs in $\mathscr{C}$ against $t(n^\delta)/q(n)$-time adversaries.*

Proof. (sketch.) The theorem follows in the same manner as Theorem 6.6.4. The proof is outlined as follows:

First, we construct a collection of weak PPRGs in $\mathscr{C}^k$ (for some $k \in \mathbb{N}$) based on the hardness assumption of learning. Then, we apply Theorem 6.6.9 to convert the weak PPRG into a standard PPRG in $\mathscr{C}$, where we use the junta-composition condition to apply Theorem 6.6.9. For the collection of weak PPRGs, we apply the same construction as Theorem 6.6.4, i.e., each bit of the generator takes the form of x^* for some $x \in \{0,1\}^*$, where x is an example selected according to the example distribution in the hard learning problem (note that the difference with Theorem 6.6.4 is that x is not always sparse in this case). By the FPT dualization in $\mathscr{C}^k$, each bit of the generator is computable in $\mathscr{C}^k$ for some $k \in \mathbb{N}$ when the description of the dual of a target function is given as the input. Remember that a target function in the hard learning problem is samplable in $\mathscr{C}^{k'}$ for some $k' \in \mathbb{N}$, and the dual of the target function is computable in $\mathscr{C}^{k''}$ for some $k'' \in \mathbb{N}$. Therefore, by considering the composite functions of these three types of functions, we can construct a generator whose input is the random seed for selecting a target function. By the junta-composition condition, each bit of the generator is computable in $\mathscr{C}^{k'''}$ for some $k''' \in \mathbb{N}$. □

Theorem 6.6.2 holds by applying Theorems 6.6.10 and 6.6.11 for all polynomial time-bounds $t(n)$.

6.7 Learning vs. PPRG in Constant-Degree Polynomials

We show Theorem 6.2.3 in Section 6.7.1 and Theorem 6.2.4 in Section 6.7.2, which are formally stated as follows.

Theorem 6.7.1. *For any polynomial $p(n), r(n)$, the following are equivalent:*

1. *There exists a PPRG in $\oplus$-NC^0.*
2. *Degree-d $\mathbb{F}_2$-polynomials are not polynomial-time learnable on average with advantage $1/2 - 1/p(n)$ with respect to a uniform example distribution and a target distribution samplable by a degree-d' $\mathbb{F}_2$-polynomial using $r(n)$-bit random seeds with (d,d')-FPT samples.*

Theorem 6.7.2. *For any polynomial $p(n), r(n)$, the following are equivalent:*

1. *There exists a collection of PPRGs in $\oplus$-NC^0.*
2. *Degree-d $\mathbb{F}_2$-polynomials are not polynomial-time learnable on average with advantage $1/2 - 1/p(n)$ with respect to an example distribution samplable with shared randomness and a target distribution samplable by a degree-d' $\mathbb{F}_2$-polynomial using $r(n)$-bit random seeds with (d, d')-FPT samples.*

6.7.1 PPRG vs. Learning on Uniform Example Distribution

In this section, we show the equivalence between a PPRG in constant-degree $\mathbb{F}_2$-polynomials (i.e., $\oplus$-NC^0) and average-case hardness of learning constant-degree $\mathbb{F}_2$-polynomials with respect to the uniform example distribution and a target distribution samplable by constant-degree $\mathbb{F}_2$-polynomials. First, we show the following key lemma.

Lemma 6.7.3. *For any $d \in \mathbb{N}$, let $p\colon \mathbb{F}_2^n \to \mathbb{F}_2$ denote a random degree-d $\mathbb{F}_2$-polynomial, i.e., each coefficient of p is selected uniformly at random from $\{0,1\}$. Let $m(n) = 1/2 \cdot \binom{n}{\le d}$. Then,*

$$\left((U_n^{(1)}, p(U_n^{(1)})), \dots, (U_n^{(m(n))}, p(U_n^{(m(n))}))\right) \equiv_s \left((U_n^{(1)}, U_1^{(1)}), \dots, (U_n^{(m(n))}, U_1^{(m(n))})\right).$$

Proof. We identify $\left[\binom{n}{\le d}\right]$ with $\{S \subseteq [n] : |S| \le d\}$ in some canonical ordering. For $\ell \in \mathbb{N}$ and $x^{(1)}, \dots, x^{(\ell)} \in \mathbb{F}_2^n$, we define a matrix $A[x^{(1)}, \dots, x^{(\ell)}] \in \mathbb{F}_2^{\ell \times \binom{n}{\le d}}$ as $A[x^{(1)}, \dots, x^{(\ell)}]_{i,S} = \prod_{j \in S} x_j^{(i)}$ for each $i \in [\ell]$ and $S \subseteq [n]$ with $|S| \le d$.

We identify a degree-d $\mathbb{F}_2$-polynomial p with a string in $\mathbb{F}_2^{\binom{n}{\le d}}$ consisting of coefficients. Then, for each $x^{(1)}, \dots, x^{(m)} \in \mathbb{F}_2^n$ and each n-input degree-d $\mathbb{F}_2$ polynomial p, the vector $[p(x^{(1)}), \dots, p(x^{(m)})]^T$ is represented as $A[x^{(1)}, \dots, x^{(m)}] \cdot p$.

Now, we assume that $x^{(1)}, \dots, x^{(m)}$ satisfy that $A[x^{(1)}, \dots, x^{(m)}]$ has full rank. Then, there exists a full-rank matrix $B \in \mathbb{F}_2^{m \times m}$ such that

$$I := [e^1, \dots, e^m, *, \dots, *] = B \cdot A[x^{(1)}, \dots, x^{(m)}],$$

where $e^1, \dots, e^m \in \mathbb{F}_2^m$, and each e^i is the unit vector (i.e., $e^i_j = 1$ iff $i = j$). In this case, we have

$$A[x^{(1)}, \dots, x^{(m)}] \cdot p = B^{-1} I \cdot p = B^{-1} \cdot \left[p_1 + f^1\left(p_{m+1}, \dots, p_{\binom{n}{\le d}}\right), \dots, p_m + f^m\left(p_{m+1}, \dots, p_{\binom{n}{\le d}}\right)\right]^T,$$

for some functions $f^1, \dots, f^m$. Since B^{-1} has full rank, if p is selected uniformly at random, then $[p(x^{(1)}), \dots, p(x^{(m)})]^T$ is also distributed uniformly at random over the choice of p. Thus, it is sufficient to show that the probability that $A[x^{(1)}, \dots, x^{(m)}]$ does not have full rank is negligible over the choices of $x^{(1)}, \dots, x^{(m)}$.

Fix $i \in [m]$ arbitrarily. Suppose that we have selected $x^{(1)}, \dots, x^{(i-1)}$ such that $A[x^{(1)}, \dots, x^{(i-1)}]$ has full rank, and we select a new $x^{(i)} \in \mathbb{F}_2^n$ uniformly at random. Then, we show that the conditional probability that $A[x^{(1)}, \dots, x^{(i)}]$ also has full rank with probability at least $1 - 2^{-\Omega(n)}$. If this is correct, then by the union bound, the probability that $A[x^{(1)}, \dots, x^{(m)}]$ does not have full rank is bounded above by $m \cdot 2^{-\Omega(n)} = O(n^d) \cdot 2^{-\Omega(n)} = \mathsf{negl}(n)$ because there must exist $i \in [m]$ such that $A[x^{(1)}, \dots, x^{(i)}]$ does not have full rank in such a case.

Let $V \leq \mathbb{F}_2^{\binom{n}{\leq d}}$ be the linear subspace spanned by the rows in $A[x^{(1)}, \ldots, x^{(i-1)}]$. Note that $\dim V \leq i-1$. For each $x \in \mathbb{F}_2^n$, we define $\tilde{x} \in \mathbb{F}_2^{\binom{n}{\leq d}}$, where $\tilde{x}_S = \prod_{j \in S} x_j$ for each $S \subseteq [n]$ with $|S| \leq d$. Let $U = \{\tilde{x} : x \in \mathbb{F}_2^n\}$. Then, it is not hard to verify that

$$\Pr_{x^{(i)}}\left[A[x^{(1)}, \ldots, x^{(i)}] \text{ does not have full rank} \middle| x^{(1)}, \ldots, x^{(i-1)}\right] = \frac{|V \cap U|}{|U|}.$$

Let $k = \lfloor \log |V \cap U| \rfloor$. Then, we have $2^k \leq |V \cap U| \leq 2^{k+1}$. We fix 2^k distinct elements $y^{(1)}, \ldots, y^{(2^k)} \in \mathbb{F}_2^n$ such that $\tilde{y}^{(1)}, \ldots, \tilde{y}^{(2^k)} \in V \cap U$. Let $V' = \mathrm{span}\{y^{(1)}, \ldots, y^{(2^k)}\}$. Then, V' is a linear subspace of V. By Lemma 6.5.2, we have

$$\binom{k}{\leq d} \leq \dim V' \leq \dim V \leq i-1 \leq m = \frac{1}{2}\binom{n}{\leq d}.$$

By Lemma 6.5.1, there exists a constant $\gamma \in (0,1)$ such that $k \leq n(1-\gamma/d)$ for any sufficiently large $n \in \mathbb{N}$. Therefore, we conclude that

$$\begin{aligned}
\Pr_{x^{(i)}}\left[A[x^{(1)}, \ldots, x^{(i)}] \text{ does not have full rank} \middle| x^{(1)}, \ldots, x^{(i-1)}\right] &= \frac{|V \cap U|}{|U|} \\
&\leq \frac{2^{k+1}}{2^n} \\
&\leq \frac{2 \cdot 2^{n(1-\gamma/d)}}{2^n} \\
&= 2^{-\frac{\gamma}{d}n+1} \\
&= 2^{-\Omega(n)}
\end{aligned}$$

□

We remark that Lemma 6.7.3 implies that learning degree-d $\mathbb{F}_2$-polynomials is infeasible with $2^{-1} \cdot \binom{n}{\leq d} = \Omega(n^d)$ samples and non-negligible advantage even for time-unbounded learners with respect to the uniform example distribution and the uniform target distribution over degree-d $\mathbb{F}_2$-polynomials. In this sense, the upper bound on the seed length for a target distribution is essential in Theorems 6.7.1 and 6.7.2.

We now show one direction from PPRGs to the average-case hardness of learning.

Theorem 6.7.4. *For any polynomial $p(n)$, there exists a polynomial q such that for any time-bound function $t(n)$, if there exists a PPRG computable by constant-degree $\mathbb{F}_2$-polynomials against $t(n)$-time adversaries, then degree-d $\mathbb{F}_2$-polynomials are not $(t(n)/q(n), 1/p(n))$-learnable on average with respect to a uniform example distribution and a target distribution samplable by a degree-d' $\mathbb{F}_2$-polynomial using an n-bit random seed with (d, d')-FPT samples.*

Proof. We assume that G is a PPRG computable by degree-d_0 polynomials for some $d_0 \in \mathbb{N}$. Fix an FPT sample-complexity function $m_{d,d'}(n) = f(d,d') \cdot p_m(n)$ arbitrarily, where $p_m(n)$ is a polynomial. We select $d \in \mathbb{N}$ such that $1/2 \cdot \binom{n}{\leq d} > \log n \cdot p_m(n)$.

We consider a pseudorandom degree-d $\mathbb{F}_2$ polynomial $p_{PR} : \mathbb{F}_2^n \to \mathbb{F}_2$, where each coefficient is selected by a pseudorandom string $G^{(d)}(U_n)$, where $G^{(d)}$ is the PPRG in Theorem 6.5.11. Then, we specify a target distribution $\mathcal{D}_{targ}$ for the hard problem as the distribution of p_{PR}. Since each

pseudorandom bit of $G^{(d)}(U_n)$ is computable by a degree-d' $\mathbb{F}_2$-polynomial for some $d' \in \mathbb{N}$, the target distribution $\mathcal{D}_{targ}$ is samplable by the degree-d' $\mathbb{F}_2$-polynomial using an n-bit random seed.

We prove the theorem by contradiction. Suppose that there exists a $t(n)$-time algorithm L that learns degree-d $\mathbb{F}_2$-polynomial with respect to the uniform example distribution and $\mathcal{D}_{targ}$ with $m_{d,d'}(n)$ samples. Because no learning algorithm can guess a random function non-negligibly better than a random guess, L is converted into a distinguisher D with advantage $1/p(n) - \mathsf{negl}(n)$ for the following two distributions: (1) $(U_n^{(1)}, p_{PR}(U_n^{(1)})), \dots, (U_n^{(m_{d,d'}(n))}, p_{PR}(U_n^{(m_{d,d'}(n))}))$ and (2) $(U_n^{(1)}, U_1^{(1)}), \dots, (U_n^{(m_{d,d'}(n))}, U_1^{(m_{d,d'}(n))})$. Since $f(d,d') \cdot p_m(n) < \log n \cdot p_m(n) < 1/2 \cdot \binom{n}{\leq d}$ for sufficiently large $n \in \mathbb{N}$, by Lemma 6.7.3, D distinguishes (1) $(U_n^{(1)}, p_{PR}(U_n^{(1)})), \dots, (U_n^{(m_{d,d'}(n))}, p_{PR}(U_n^{(m_{d,d'}(n))}))$ and (2') $(U_n^{(1)}, p_R(U_n^{(1)})), \dots, (U_n^{(m_{d,d'}(n))}, p_R(U_n^{(m_{d,d'}(n))}))$, where p_R is a truly random degree-d $\mathbb{F}_2$-polynomial. Then, we can construct an $O(m_{d,d'}(n) + t(n))$-time adversary A for $G^{(d)}$ that is given a pseudorandom or random string r, selects a degree-d polynomial p_r by using r, makes $m_{d,d'}(n)$ samples for p_r for random inputs, and feeds them to D. By Theorem 6.5.11, A is converted into an adversary A' breaking G. It is not hard to verify that the running time is bounded above by $\mathsf{poly}(n) \cdot O(m_{d,d'}(n) + t(n)) \leq q(n) \cdot t(n)$ for some polynomial q. □

Next, we show the opposite direction, i.e., from the average-case hardness of learning to PPRGs. Theorem 6.7.1 is obtained by applying Theorems 6.7.4 and 6.7.5 for all polynomial time-bounds $t(n)$.

Theorem 6.7.5. *For any polynomial $p(n), r(n)$, there exist a polynomial q and a constant $\epsilon > 0$ such that for any time-bound function $t(n)$, if degree-d polynomials are not $(t(n), 1/2 - 1/p(n))$-learnable on average with respect to a uniform example distribution and a target distribution samplable by a degree-d' $\mathbb{F}_2$-polynomial using an $r(n)$-bit random seed with (d,d')-FPT samples, then there exists a PPRG computable by a constant-degree $\mathbb{F}_2$-polynomial against $t(n^\epsilon)/q(n)$-time adversaries.*

Proof. Let $m(n)$ and $\ell(n)$ be the polynomials obtained by applying Fact 6.5.8 for $p(n)$ and $m^{\oplus}(n) = \ell(n)^2(n + r(n))^2$ (note that these are well-defined because $\ell(n)$ is determined by only $p(n)$). Then, based on the hardness assumption, there exist constants $d, d' \in \mathbb{N}$ and a target distribution $\mathcal{D}_{targ}$ on degree-d $\mathbb{F}_2$-polynomials for the hard learning problem with $m(n)$ samples and advantage $1/2 - 1/p(n)$, where $\mathcal{D}_{targ}$ is samplable by a degree-d' $\mathbb{F}_2$-polynomial using an $r(n)$-bit random seed. Let $p_{targ} \colon \{0,1\}^{r(n)} \to \{0,1\}^{\binom{n}{\leq d}}$ be the degree-d' $\mathbb{F}_2$-polynomials that select a target function according to $\mathcal{D}_{targ}$.

For $R(n) = \ell(n)(n + r(n))$, we construct a PPRG $G \colon \{0,1\}^{R(n)^3} \to \{0,1\}^{R(n)^4}$. We remark that G is defined on only the input length $R(n)^3$. For a PPRG on any input length, we apply the following simple technique. For a given n-bit random seed, we find the maximum integer n' such that $R(n')^3 \leq n$, separate the seed into blocks of size n', and apply the original PRG to each block (where the remaining seed of length $n - n' \cdot \lfloor n/n' \rfloor$ is outputted directly). For the security proof, we apply the standard hybrid argument (for details, refer to [Gol01]). Since $R(n)^3$ is a polynomial in n, it is not hard to verify that the resulting PRG still preserves polynomial-stretch.

Now, we present the construction of G for input length $R(n)^3$. For convenience, we assume that the random seed $R(n)^3$ is separated into $\ell(n)R(n)^2$ strings $x^{i,j} \in \{0,1\}^n$ and $y^{i,j} \in \{0,1\}^{r(n)}$ indexed by $(i,j) \in [R^2(n)] \times [\ell(n)]$. First, $G(x,y)$ generates target functions $f^{i,j} = p_{targ}(y^{i,j})$ for each i, j. Then, G computes $b^{i_1,i_2} := \bigoplus_{j_1,j_2 \in [\ell(n)]} f^{i_1,j_1}(x^{i_2,j_2})$ for all $i_1, i_2 \in [R(n)^2]$ and outputs them as a pseudorandom string of length $R(n)^2 \cdot R(n)^2 = R(n)^4$.

In the following, we show that (i) each b^{i_1,i_2} is computed by a constant-degree $\mathbb{F}_2$-polynomial, and (ii) the above-mentioned G is a pseudorandom generator, which implies the theorem.

For statement (i), we remark that for any depth-d $\mathbb{F}_2$-polynomial f^{i_1,j_1} and the input x^{i_2,j_2}, the value of $f^{i_1,j_1}(x^{i_2,j_2})$ is computable by depth-$(d+1)$ $\mathbb{F}_2$-polynomial given f^{i_1,j_1} and x^{i_2,j_2} as input because $f^{i_1,j_1}(x^{i_2,j_2}) = \sum_{S:|S|\leq d} f_S^{i_1,j_1} x_S^{i_2,j_2}$. Furthermore, each bit of f^{i_1,j_1} is computable by degree-d' $\mathbb{F}_2$-polynomials in y^{i_1,j_1}. Therefore, each $f^{i_1,j_1}(x^{i_2,j_2})$ is computable by an $\mathbb{F}_2$-polynomial of degree $d'(d+1)$, and so is $b^{i_1,i_2} := \bigoplus_{j_1,j_2\in[n]} f^{i_1,j_1}(x^{i_2,j_2})$.

Next, we show statement (ii) by contradiction. The outline of the proof is as follows. First, by assuming that there exists an adversary A that breaks G with non-negligible advantage, we show that degree-d $\mathbb{F}_2$-polynomials are learnable on $\mathcal{D}_{ex}^{\oplus}$ and $\mathcal{D}_{targ}^{\oplus}$ with $R(n)^2 - 1$ samples and non-negligible advantage, where $\mathcal{D}_{ex}^{\oplus}$ and $\mathcal{D}_{targ}^{\oplus}$ are the distributions of $x^{\oplus} = x^{(1)} \circ \cdots \circ x^{(\ell(n))}$ and $f^{\oplus}(x^{\oplus}) := \bigoplus_{i,j} f^{(i)}(x^{(j)})$ for $x^{(1)}, \ldots, x^{(\ell(n))} \sim U_n$ and $f^{(1)}, \ldots, f^{(\ell(n))} \sim \mathcal{D}_{targ}$, respectively. Then, by applying the XOR lemma (i.e., Fact 6.5.8), we show that degree-d $\mathbb{F}_2$-polynomials are learnable with respect to U_n and $\mathcal{D}_{targ}$ with $m(n)$ samples and an advantage of $1/2 - 1/p(n)$, which contradicts the hardness assumption of learning.

For sufficiently large $n \in \mathbb{N}$, we assume that[9]

$$\Pr\left[A(G(U_{R(n)^3})) = 1\right] - \Pr\left[A(U_{R(n)^4}) = 1\right] \geq 1/\mathsf{poly}(n). \tag{6.2}$$

We construct a learner $L^{\oplus}$ on $\mathcal{D}_{ex}^{\oplus}$ and $\mathcal{D}_{targ}^{\oplus}$ as follows. On input $(x^{(1)}, b^{(1)}), \ldots, (x^{(R(n)^2)}, b^{(R(n)^2)})$ and a challenge x_c, where each $x^{(i)}$ consists of $x^{i,1}, \ldots, x^{i,\ell(n)} \sim U_n$, the learner $L^{\oplus}$ randomly selects $I_1, I_2 \sim [R(n)^2]$ and $f^{i,j} \sim \mathcal{D}_{targ}$ for each $i < I_1$ and $j \in [\ell(n)]$. For simplicity, $L^{\oplus}$ changes the indices as $x^{(I_2)} := x_c$ and $(x^{(i)}, b^{(i)}) := (x^{(i+1)}, b^{(i+1)})$ for each $i > I_2$, i.e., $L^{\oplus}$ inserts the challenge in the I_2-th position in samples. Then, $L^{\oplus}$ executes $A(b^{1,1}, \ldots, b^{R(n)^2,R(n)^2})$, where each b^{i_1,i_2} is defined by

$$b^{i_1,i_2} = \begin{cases} \bigoplus_{j_1,j_2\in[\ell(n)]} f^{i_1,j_1}(x^{i_2,j_2}) & \text{if } i_1 < I_1 \\ b^{(i)} & \text{if } (i_1 = I_1) \wedge (i_2 < I_2) \\ r^{i_1,i_2} & \text{if } (i_1 = I_1) \wedge (i_2 \geq I_2) \\ r^{i_1,i_2} & \text{if } i_1 > I_1, \end{cases}$$

where $r^{i_1,i_2} \sim \{0,1\}$. If A outputs 1, then $L^{\oplus}$ outputs r^{I_1,I_2} as a prediction; otherwise, $1 - r^{I_1,I_2}$.

We show the correctness of $L^{\oplus}$. For each I_1 and I_2, we define the hybrid distribution H_{I_1,I_2} as the distribution of $b^{1,1}, \ldots, b^{R(n)^2,R(n)^2}$ selected as

$$b^{i_1,i_2} = \begin{cases} \bigoplus_{j_1,j_2\in[n]} f^{i_1,j_1}(x^{i_2,j_2}) & \text{if } i_1 < I_1 \\ \bigoplus_{j_1,j_2\in[n]} f^{i_1,j_1}(x^{i_2,j_2}) & \text{if } (i_1 = I_1) \wedge (i_2 \leq I_2) \\ r^{i_1,i_2} & \text{if } (i_1 = I_1) \wedge (i_2 > I_2) \\ r^{i_1,i_2} & \text{if } i_1 > I_1, \end{cases}$$

where $f^{i_1,j_1} \sim \mathcal{D}_{targ}$, $x^{i_2,j_2} \sim \mathcal{D}_{ex}$, and $r^{i_1,i_2} \sim \{0,1\}$. Then, it is easily verified that $H_{1,0} \equiv U_{R(n)^4}$, $H_{R(n)^2,R(n)^2} \equiv G(U_{R(n)^3})$, and $H_{I_1,0} \equiv H_{I_1-1,R(n)^2}$ for each $I_1 \in [R(n)^2]$. Therefore, by Eq. (6.2), we have

$$\Pr[A(H_{R(n)^2,R(n)^2}) = 1] - \Pr[A(H_{1,0}) = 1] \geq 1/\mathsf{poly}(n).$$

[9]Strictly speaking, we test the behavior of the adversary first, then take a negation according to the result to remove the vertical bars for an absolute value.

For each I_1, I_2 selected by $L^{\oplus}$, the probability that $L^{\oplus}$ outputs the correct prediction b_c is

$$\begin{aligned}\Pr[r^{I_1,I_2} = b_c]\Pr[A(H_{I_1,I_2}) = 1] + \Pr[r^{I_1,I_2} = 1 - b_c]\Pr[A(\bar{H}_{I_1,I_2}) = 0] \\ = \frac{1}{2} + \frac{1}{2}\Pr[A(H_{I_1,I_2}) = 1] - \frac{1}{2}\Pr[A(\bar{H}_{I_1,I_2}) = 1],\end{aligned}$$

where $\bar{H}_{I_1,I_2}$ is the same distribution as H_{I_1,I_2} except that the (I_1, I_2)-th bit is flipped.

We remark that

$$\Pr[A(H_{I_1,I_2-1}) = 1] = \frac{1}{2}\Pr[A(H_{I_1,I_2}) = 1] + \frac{1}{2}\Pr[A(\bar{H}_{I_1,I_2}) = 1].$$

Thus, the probability that $L^{\oplus}$ succeeds in predicting b_c conditioned on I_1, I_2 is

$$\begin{aligned}&\frac{1}{2} + \frac{1}{2}\Pr[A(H_{I_1,I_2}) = 1] - \frac{1}{2}\Pr[A(\bar{H}_{I_1,I_2}) = 1] \\ &= \frac{1}{2} + \frac{1}{2}\Pr[A(H_{I_1,I_2}) = 1] - \left(\Pr[A(H_{I_1,I_2-1}) = 1] - \frac{1}{2}\Pr[A(H_{I_1,I_2}) = 1]\right) \\ &= \frac{1}{2} + \Pr[A(H_{I_1,I_2}) = 1] - \Pr[A(H_{I_1,I_2-1}) = 1].\end{aligned}$$

Therefore, the success probability of $L^{\oplus}$ is at least

$$\begin{aligned}&\sum_{I_1,I_2\in[R(n)^2]} \frac{1}{R(n)^4}\left(\frac{1}{2} + \Pr[A(H_{I_1,I_2}) = 1] - \Pr[A(H_{I_1,I_2-1}) = 1]\right) \\ &= \frac{1}{2} + \frac{1}{R(n)^4}(\Pr[A(H_{R(n)^2,R(n)^2}) = 1] - \Pr[A(H_{1,0}) = 1]) \\ &\geq \frac{1}{2} + \frac{1}{\mathsf{poly}(n)R(n)^4} \\ &\geq \frac{1}{2} + \frac{1}{\mathsf{poly}(n)},\end{aligned}$$

where the first equality holds because $H_{I_1,0} \equiv H_{I_1-1,R(n)^2}$ for each $I_1 \in [R(n)^2]$.

Since $L^{\oplus}$ succeeds in learning on $\mathcal{D}^{\oplus}_{ex}$ and $\mathcal{D}^{\oplus}_{targ}$ with $R(n)^2-1 < R(n)^2 = m^{\oplus}(n)$ $(< m^{\oplus}(n\ell(n)))$ samples, **Boost** in Lemma 6.5.8 succeeds in learning on U_n and $\mathcal{D}_{targ}$ with $m(n)$ samples and advantage $1/2-1/p(n)$ (for sufficiently large n). It is not hard to verify that if the running time of A is bounded by $T(n)$, then the running time of the learner is at most $\mathsf{poly}(n)\cdot T(R^3(n)) \leq q(n)\cdot T(n^a)$ for a sufficiently large polynomial q and a sufficiently large constant $a \geq 1$. By letting $\epsilon = 1/a$, the theorem follows. □

6.7.2 A Collection of PPRGs vs. Learning on Example Distribution Samplable with Shared Randomness

In this section, we show the equivalence between a collection of PPRGs in constant-degree $\mathbb{F}_2$-polynomials and average-case hardness of learning constant-degree $\mathbb{F}_2$-polynomials with respect to an example distribution samplable with shared randomness and a target distribution samplable by constant-degree $\mathbb{F}_2$-polynomials. First, we introduce a useful lemma.

Lemma 6.7.6. *For $n, d, d' \in \mathbb{N}$, let $p_1, \dots, p_n \colon \{0,1\}^n \to \{0,1\}$ be degree-d $\mathbb{F}_2$-polynomials, and let $p' \colon \{0,1\}^n \to \{0,1\}$ be a degree-d' $\mathbb{F}_2$-polynomial. We define a degree-dd' $\mathbb{F}_2$-polynomial $q \colon \{0,1\}^n \to \{0,1\}$ as $q(x) := p'(p_1(x), \dots, p_n(x))$. Then, the representation of q is computed by degree-$(d'+1)$ $\mathbb{F}_2$-polynomials given $p_1, \dots, p_n, p'$ as the input.*

Proof. For each $S \subseteq [n]$ with $|S| \leq dd'$, we show that q_S is computed by a degree-$(d'+1)$ $\mathbb{F}_2$-polynomial given $p_1, \dots, p_n, p'$ as the input.

We consider the expansion of the following formula:

$$q(x) = \sum_{I \subseteq [n] : |I| \leq d'} p'_I \prod_{i \in I} p_i(x) = \sum_{I \subseteq [n] : |I| \leq d'} p'_I \prod_{i \in I} \sum_{J \subseteq [n] : |J| \leq d} (p_i)_J x^J.$$

For convenience, let $r^I(x) := p'_I \prod_{i \in I} \sum_{J \subseteq [n]} (p_i)_J x^J$ for each I, i.e., $q(x) = \sum_{I : |I| \leq d'} p'_I \cdot r^I(x)$.

Fix $I \subseteq [n]$ with $k := |I| \leq d'$ arbitrarily, and let $I = \{i_1, \dots, i_k\}$. Since $x_i^2 = x_i$ for each $i \in [n]$, by expanding r^I, it is not hard to verify that

$$(r^I)_S = \sum_{\substack{J_1, \dots, J_k \subseteq [n] : \\ J_1 \cup \dots \cup J_k = S}} (p_{i_1})_{J_1} \cdot (p_{i_2})_{J_2} \cdot \dots \cdot (p_{i_k})_{J_k}.$$

Therefore, the degree of $(r^I)_S$ as an $\mathbb{F}_2$-polynomial in $p_1, \dots, p_n, p'$ is at most $k = |I|$. Since $q_S = \sum_{I : |I| \leq d'} p'_I \cdot (r^I)_S$, the degree of q_S is at most $d'+1$ as an $\mathbb{F}_2$-polynomial in $p_1, \dots, p_n, p'$. □

Now, we show one direction from a collection of PPRGs to the average-case hardness of learning.

Theorem 6.7.7. *For any polynomial $p(n)$, there exist a polynomial q and a constant $\epsilon \in (0,1)$ such that for any time-bound function $t(n)$, if there exists a collection of PPRGs in constant-degree $\mathbb{F}_2$-polynomials against $t(n)$-time adversaries, then degree-d $\mathbb{F}_2$-polynomials are not $(t(n)/q(n), 1/p(n))$-learnable on average with respect to an example distribution samplable with shared randomness and a target distribution samplable by a degree-d' $\mathbb{F}_2$-polynomial using an n-bit random seed with (d, d')-FPT samples.*

Proof. We assume that G is a collection of PPRGs in degree-d_0 polynomials for some $d_0 \in \mathbb{N}$. Without loss of generality, we can assume that the output length of G is n^2; otherwise, we regard $G^{(1)}$ in Theorem 6.5.11 as G. Fix an FPT sample-complexity function $m_{d,d'}(n) = f(d,d') \cdot p_m(n)$ arbitrarily, where $p_m(n)$ is a polynomial. We select $d \in \mathbb{N}$ such that $n^{d+1} > \log n \cdot p_m(n^{d_0+1} \cdot \log n) \cdot p(n^{d_0+1} \cdot \log n)$.

Now, we specify an example distribution $\mathcal{D}_{ex}$ and a target distribution $\mathcal{D}_{targ}$ for the hardness of learning with sample complexity $m_{d,d'}$. We define $\mathcal{D}_{ex}$ as the distribution of the concatenation of binary representations $G_{i_1 \cdot n+1}, \dots, G_{i_1 \cdot n+n}, \dots, G_{i_{d-1} \cdot n+1}, \dots, G_{i_{d-1} \cdot n+n}, G_{i_d}$, where $i_1, \dots, i_{d-1} \sim [n-1] \cup \{0\}$ and $i_d \sim [n^2]$ (where the choice of G is simulated by shared randomness). We remark that the following composition of the selected polynomials

$$G_{i_1, \dots, i_d}(x) := G_{i_d}(G_{i_{d-1}+1}(\cdots G_{i_2 \cdot n+1}(G_{i_1 \cdot n+1}(x), \dots, G_{i_1 \cdot n+n}(x)), \dots,), \dots, G_{i_{d-1}+n}(\dots))$$

is distributed according to the uniform distribution over $G_1^{(d)}, \dots, G_{n^{d+1}}^{(d)}$.

The outline of the remaining proof is as follows. First, we show that (i) for any $x \in \{0,1\}^n$, there exists a constant-degree $\mathbb{F}_2$-polynomial x^* such that $x^*(G_{i_1 \cdot n+1}, \dots, G_{i_d}) = G_{i_1, \dots, i_d}(x)$ for any choice of $i_1, \dots, i_d$. Then, we define the distribution $\mathcal{D}_{targ}$ as the distribution of x^* for $x \sim \{0,1\}^n$

and show that (ii) $\mathcal{D}_{targ}$ is samplable by constant-degree $\mathbb{F}_2$-polynomials that takes x as the input (i.e., a random seed). Finally, by a similar proof as Theorem 6.6.3, we show that (iii) learning constant-degree $\mathbb{F}_2$-polynomials is hard with respect to $\mathcal{D}_{ex}$ and $\mathcal{D}_{targ}$.

By induction in d, we show that for any $x \in \{0,1\}^n$, there exists a degree-$(d_0+1)^{d-1}$ $\mathbb{F}_2$-polynomial that is given $G_{i_1 \cdot n+1}, \ldots, G_{i_d}$ and outputs $G_{i_1,\ldots,i_d}$. The base step (i.e., $d = 1$) is trivial. For the inductive step, we assume that the claim holds in the case of $d-1$. Then, for any $i \in [n]$, there exists a degree-$(d_0+1)^{d-2}$ $\mathbb{F}_2$-polynomial q_i that is given $G_{i_1 \cdot n+1}, \ldots, G_{i_{d-2} \cdot n+n}, G_{i_{d-1} \cdot n+i}$ and outputs $G_{i_1,\ldots,i_{d-1}+i}$ for each $i_1, \ldots, i_{d-1}$. Now, we apply Lemma 6.7.6 for $p_1 = G_{i_1,\ldots,i_{d-1}+1}, \ldots, p_n = G_{i_1,\ldots,i_{d-1}+n}$, and $p' = G_{i_d}$. Then, the degree-(d_0+1) polynomial q in Lemma 6.7.6 outputs $G_{i_1,\ldots,i_d}$ for given $G_{i_1,\ldots,i_{d-1}+1}, \ldots, G_{i_d}$. Since each bit of $G_{i_1,\ldots,i_{d-1}+i}$ is computed by q_i, this q is implemented by an $\mathbb{F}_2$-polynomial in $G_{i_1 \cdot n+1}, \ldots, G_{i_d}$, where the degree is at most $(d_0+1)^{d-2} \cdot (d_0+1) = (d_0+1)^{d-1}$.

This claim implies statement (i) for the following reason. Since the degree of $G_{i_1,\ldots,i_d}(x)$ is at most d_0^d, it is written as

$$G_{i_1,\ldots,i_d}(x) = \sum_{S:|S| \le d_0^d} (G_{i_1,\ldots,i_d})_S x^S.$$

We can regard this expression as a degree-1 $\binom{n}{\le d_0^d}$-input $\mathbb{F}_2$-polynomial in $G_{i_1,\ldots,i_d}$. By the claim above, each bit of the representation of $G_{i_1,\ldots,i_d}$ is computed by degree-$(d_0+1)^{d-1}$ $\mathbb{F}_2$-polynomials in $G_{i_1 \cdot n+1}, \ldots, G_{i_d}$. Thus, $G_{i_1,\ldots,i_d}(x)$ is computed by a degree-$(d_0+1)^{d-1}$ $\mathbb{F}_2$-polynomial $x^*(G_{i_1 \cdot n+1}, \ldots, G_{i_d})$, which is determined only by x. We remark that, by the argument above, we reduce the input size from $O(n^{d_0^d})$ to $O(d \cdot n^{d_0}) \le n^{d_0} \cdot \log n$ at the expense of the degree of the target function.

Next, we show statement (ii), i.e., the target distribution $\mathcal{D}_{targ}$ of x^* for $x \sim \{0,1\}^n$ is samplable by degree-d_0^d $\mathbb{F}_2$-polynomials. Based on the argument above, the polynomial x^* is represented as

$$x^*(G_{i_1 \cdot n+1}, \ldots, G_{i_d}) = \sum_{S:|S| \le d_0^d} p^S(G_{i_1 \cdot n+1}, \ldots, G_{i_d}) x^S,$$

where p^S is some polynomial of degree $(d_0+1)^{d-1}$ for each $S \subseteq [n]$ with $|S| \le d_0^d$. Thus, for each $T \subseteq \left[((d-1)n+1)\binom{n}{\le d_0}\right]$ (note that $((d-1)n+1)\binom{n}{\le d_0}$ is the input length of x^*), the coefficient $(x^*)_T$ is written as

$$(x^*)_T = \sum_{S:|S| \le d_0^d} (p^S)_T \cdot x^S.$$

Therefore, x^* is computed by an $\mathbb{F}_2$-polynomial (given x as the input) of degree d_0^d, and $\mathcal{D}_{targ}$ is samplable by $\mathbb{F}_2$-polynomials of degree d_0^d, where the seed length is $|x| = n$.

Finally, we prove statement (iii) by contradiction. Suppose that there exists an algorithm L that succeeds in $(t(n), 1/p(n))$-learning $\mathbb{F}_2$-polynomials with sample complexity $m(n) := m_{(d_0+1)^{d-1}, d_0^d}(n)$. Then, we construct an adversary A for $G^{(d)}$ based on L, which also yields an adversary for G.

On input $w \in \{0,1\}^{n^{d+1}}$ and a description of G, where w is a pseudorandom string generated by $G^{(d)}$ or a truly random string, A simulates the example distribution $\mathcal{D}_{ex}$ by selecting $G_{i_1 \cdot n+1}, \ldots, G_{i_d}$ for $i_1, \ldots, i_{d-1} \sim [n-1] \cup \{0\}$ and $i_d \sim [n^2]$. For convenience, we let N denote the size of each example, i.e., $N := ((d-1)n+1)\binom{n}{\le d_0} = O(d \cdot n^{d_0+1})$. After generating $m(N)$ samples, A also generates a challenge according to $\mathcal{D}_{ex}$ and feeds them to L. Let $i_c \in [n^{d+1}]$ be the position in $G^{(d)}$ that corresponds to the challenge. If L outputs some prediction $b \in \{0,1\}$, then L' checks whether $b = w_{i_c}$. If so, L' outputs 1; otherwise, it outputs 0.

In the case in which $w \sim G^{(d)}(x)$ for $x \sim \{0,1\}^n$, we have $x^*(G_{i_1 \cdot n+1}, \ldots, G_{i_d}) = G_i^{(d)}(x) = w_i$ for each $(i_1, \ldots, i_d)$ and the corresponding position $i \in [n^{d+1}]$. Therefore, the simulated samples are valid for the target function x^*. Since A executes L on the example distribution $\mathcal{D}_{ex}$ and target distribution $\mathcal{D}_{targ}$, we have

$$\Pr_{A,G,U_n}[A(G^{(d)}(U_n)) = 1] = \Pr_{L,\mathcal{D}_{ex},\mathcal{D}_{targ}}[L \text{ succeeds in learning}] \geq \frac{1}{2} + \frac{1}{p(N)}.$$

By contrast, in the case in which $w \sim \{0,1\}^{n^{d+1}}$, the labels in the simulated samples are selected truly at random. Because no learning algorithm can guess a random label not contained in the given samples better than a random guess, i.e., with a success probability of $1/2$, we have

$$\begin{aligned}
\Pr_{A,U_{n^{d+1}}}[A(U_{n^{d+1}}) = 1] &= \Pr_{L,\mathcal{D}_{ex}}[L \text{ succeeds in learning}] \\
&\leq \frac{1}{2} \cdot \left(1 - \frac{m(N)}{n^{d+1}}\right) + 1 \cdot \frac{m(N)}{n^{d+1}} \\
&\leq \frac{1}{2} + \frac{m(n^{d_0+1} \log n)}{2n^{d+1}} \\
&\leq \frac{1}{2} + \frac{f((d_0+1)^{d-1}, d_0^d) \cdot p_m(n^{d_0+1} \log n)}{2n^{d+1}} \\
&\leq \frac{1}{2} + \frac{n^{d+1} \cdot f((d_0+1)^{d-1}, d_0^d)}{2 \log n \cdot n^{d+1} \cdot p(n^{d_0+1} \cdot \log n)} \\
&\leq \frac{1}{2} + \frac{1}{2p(n^{d_0+1} \cdot \log n)} \\
&\leq \frac{1}{2} + \frac{1}{2p(N)},
\end{aligned}$$

for sufficiently large n. Therefore, the advantage of A is at least

$$\left(\frac{1}{2} + \frac{1}{p(N)}\right) - \left(\frac{1}{2} + \frac{1}{2p(N)}\right) = \frac{1}{2p(N)},$$

and A succeeds in breaking $G^{(d)}$.

By Theorem 6.5.11, we can construct an adversary for G. It is not hard to verify that the running time is bounded above by $t(N) \cdot q(n) \leq t(n^a) \cdot q(n)$ for a sufficiently large polynomial q and a sufficiently large constant $a \geq 1$. For the theorem, we let $\epsilon = 1/a$. We remark that for input size $N = ((d-1)n+1)\binom{n}{\leq d_0}$, the length of the random seeds for $\mathcal{D}_{targ}$ is at most $n \leq N$. □

Next, we show the opposite direction from the average-case hardness of learning to a collection of PPRGs. We obtain Theorem 6.7.2 by applying Theorems 6.7.7 and 6.7.8 for all polynomial time-bounds $t(n)$.

Theorem 6.7.8. *For any polynomial $p(n), r(n)$, there exists a polynomial q and a constant $\epsilon > 0$ such that for any time-bound function $t(n)$, if degree-d $\mathbb{F}_2$-polynomials are not $(t(n), 1/2 - 1/p(n))$-learnable on average with respect to an example distribution samplable with shared randomness and a target distribution samplable by degree-d' $\mathbb{F}_2$-polynomials using an $r(n)$-bit random seed with (d, d')-FPT samples, then there exists a collection of PPRGs computable by a constant-degree $\mathbb{F}_2$-polynomial against $t(n^\epsilon)/q(n)$-time adversaries.*

Proof. (sketch.) The construction of a collection of PPRGs mainly follows the construction of the PPRG in the proof of Theorem 6.7.5.

Let $m(n)$ and $\ell(n)$ be the polynomials obtained by applying Fact 6.5.8 for $p(n)$ and $m^{\oplus}(n) = (\ell(n)r(n))^{1+\delta}$, where $\delta > 0$ is an arbitrary constant. Then, based on the hardness assumption, there exist constants $d, d' \in \mathbb{N}$, an example distribution $\mathcal{D}_{ex}$, and a target distribution $\mathcal{D}_{targ}$ on degree-d $\mathbb{F}_2$-polynomial for the hard learning problem with $m(n)$ samples and advantage $1/2 - 1/p(n)$, where $\mathcal{D}_{ex}$ is samplable with shared randomness, and $\mathcal{D}_{targ}$ is samplable by a degree-d' $\mathbb{F}_2$-polynomial using $r(n)$-bit random seeds. Let $p_{targ} \colon \{0,1\}^{r(n)} \to \{0,1\}^{\binom{n}{\le d}}$ be the degree-d' $\mathbb{F}_2$-polynomial selecting a target function according to $\mathcal{D}_{targ}$.

For $R(n) = \ell(n)r(n)$, we construct a collection of PPRGs $G \colon \{0,1\}^{R(n)} \to \{0,1\}^{R(n)^{1+\delta}}$. For convenience, we assume that the random seed of length $R(n)$ is separated into $\ell(n)$ strings $y^j \in \{0,1\}^{r(n)}$ indexed by $j \in [\ell(n)]$. Our generator G is specified with $m(n)\ell(n)$ strings $x^{i,j} \in \{0,1\}^n$ indexed by $(i,j) \in [m(n)] \times [\ell(n)]$, which are selected according to $\mathcal{D}_{ex}$ in the selection of G. First, $G(y)$ generates target functions $f^j = p_{targ}(y^j)$ for each j. Then, G computes $b^i := \bigoplus_{j_1,j_2 \in [\ell(n)]} f^{j_1}(x^{i,j_2})$ for each $i \in [m(n)]$ and outputs them as a pseudorandom string of length $m(n) = R(n)^{1+\delta}$.

The security proof for G is almost the same as Theorem 6.7.5. Next, we verify that the generator is implemented as a collection of generators in constant-degree $\mathbb{F}_2$-polynomials. For each $i \in [m(n)]$, the i-th output bit of G is

$$b^i := \bigoplus_{j_1,j_2 \in [\ell(n)]} f^{j_1}(x^{i,j_2}) = \sum_{j_1,j_2} \sum_{S:|S| \le d} f^{j_1}_S (x^{i,j_2})^S. = \sum_{j_1,j_2} \sum_{S:|S| \le d} p^S_{targ}(y^{j_1})(x^{i,j_2})^S,$$

where p^S_{targ} represents the degree-d' polynomial computing the coefficient of f on S. Since the selector of G can select a shared randomness, G can simulate the example distribution perfectly. By hardwiring the values of x^{i,j_2} in the expression above (as a part of G), each output bit of G is computable by a degree-d' $\mathbb{F}_2$-polynomial given y as the input. Thus, we conclude that G is a collection of PPRGs in constant-degree $\mathbb{F}_2$-polynomials. □

6.8 PPRGs based on Hardness of d-LRPDT

In this section, we verify Corollary 6.2.6 based on the proofs of Theorems 6.6.4, 6.7.5, and 6.7.8. For each $d \in \mathbb{N}$, we use the notation ℓ_d to refer to the length of the binary representation of degree-d parity decision trees, i.e., $\ell_d(n) = (2^d - 1) \cdot n + 2^d$.

Corollary 6.8.1 (The first item of Corollary 6.2.6)**.** *For any $\epsilon \in (0,1)$, if d-LRPDT is $(n^{1+\epsilon}, n^{-(1+\epsilon)})$-hard on an $O(1)$-sparse example distribution samplable with shared randomness for some $d \in \mathbb{N}$, then a collection of PPRGs in NC^0 exists.*

Proof. Let $p(n) = n^{1+\epsilon}$. Then, for each $d \in \mathbb{N}$, it holds that $\ell_d(n) \le n^{1+\epsilon(1-\epsilon)/2} = p(n)^{1-\epsilon/2}$ for sufficiently large $n \in \mathbb{N}$. In the proof of Theorem 6.6.4, we use the hardness assumption of learning for the sample complexity $m(n) = p(n)^{1-\frac{(\epsilon/2)}{2}} = n^{(1+\epsilon)(1-\frac{\epsilon}{4})} = n^{1+\frac{3\epsilon}{4}-\frac{\epsilon^2}{4}} \le n^{1+\epsilon}$. Thus, by the FPT dualization of parity decision trees (i.e., Mod_2-DTs), the corollary holds. □

Corollary 6.8.2 (The second item of Corollary 6.2.6)**.** *For any $\epsilon \in (0,1)$, if d-LRPDT is $(n^{1+\epsilon}, n^{-(2+\epsilon)})$-hard on the uniform example distribution for some $d \in \mathbb{N}$, then a PPRG in $\oplus$-NC^0 exists.*

Proof. First, we observe that any depth-d parity decision tree is represented by a degree-d $\mathbb{F}_2$-polynomial as follows: Let T be an arbitrary depth-d parity decision tree. For each path $p \in \{0,1\}^d$ in T, let $\chi_1^p, \dots, \chi_d^p$ be the queried linear functions at the internal nodes on the path p, and let b_p the binary label at the leaf corresponding to p. Then, it is not hard to verify that for each input x,

$$T(x) = \sum_{p \in \{0,1\}^d} b_p \prod_{i=1}^{d} \mathbb{1}\{\chi_i^p(x) = p_i\} = \sum_{p \in \{0,1\}^d} b_p \prod_{i=1}^{d} (\chi_i^p(x) + 1 + p_i).$$

Since the degree of χ_i^p is at most 1, the depth-d parity decision tree T is expressed as a degree-d $\mathbb{F}_2$-polynomial. Furthermore, a random degree-d $\mathbb{F}_2$-polynomial corresponding to a random depth-d parity decision tree is selected by a degree-$(d+1)$ $\mathbb{F}_2$-polynomial according to the expanded expression of the above by using a $\ell_d(n)$-bit random seed. Let $r(n) = n^{1+\epsilon/16} + n$. Then, for any sufficiently large $n \in \mathbb{N}$, we can bound the seed length above by $\ell_d(n) \leq n^{1+\epsilon/16} = r(n) - n$.

Let $p(n) = n^{2+\epsilon}$. Now, we apply the proof of Theorem 6.7.5, where we slightly change the construction of the PPRG G and do not use Fact 6.5.8 (i.e., the XOR lemma). Specifically, we let G select $r(n)^{1+\epsilon/4}$ examples $x^1, \dots, x^{r(n)^{1+\epsilon/4}} \in \{0,1\}^n$ uniformly at random and $r(n)^{1+\epsilon/4}$ random degree-d parity decision trees $f^1, \dots, f^{r(n)^{1+\epsilon/4}}$ as degree-d $\mathbb{F}_2$-polynomials, and output $b^{i_1,i_2} := f^{i_1}(x^{i_2})$ for each $i_1, i_2 \in [r(n)^{1+\epsilon/4}]$. We remark that by the hybrid argument in the proof of Theorem 6.7.5, we can convert the hardness of d-LRPDT with advantage $1/p(n)$ and sample complexity $r(n)$ into a weak PPRG G that stretches an $r(n)^{2+\epsilon/4}$-bit random seed to an $r(n)^{2+\epsilon/2}$-bit pseudorandom string, and the indistinguishable parameter is $r(r^{-1}(n^{1/(2+\epsilon/4)}))^{2+\epsilon/2}/p(r^{-1}(n^{1/(2+\epsilon/4)})) = n^{(2+\epsilon/2)/(2+\epsilon/4)}/p(r^{-1}(n^{1/(2+\epsilon/4)}))$, where we interpret the upper bound on the advantage $r(n)^{2+\epsilon/2}/p(n)$ of distinguishers as a function in the seed length $r(n)^{2+\epsilon/4}$. For sufficiently large $n \in \mathbb{N}$, we have $r(n) \leq n^{1+\epsilon/8} \leq n^{1+\epsilon}$. Thus, the hardness assumption of learning satisfies the requirement on the sample complexity in Corollary 6.8.2. In addition, we have $r^{-1}(n) \geq n^{1/(1+\epsilon/8)}$, and the indistinguishable parameter is at most

$$\frac{n^{\frac{2+\frac{\epsilon}{2}}{2+\frac{\epsilon}{4}}}}{p(r^{-1}(n^{\frac{1}{2+\frac{\epsilon}{4}}}))} \leq \frac{n^{\frac{8+2\epsilon}{8+\epsilon}}}{n^{\frac{32(2+\epsilon)}{(8+\epsilon)^2}}} = n^{-\frac{32(2+\epsilon)-(8+2\epsilon)(8+\epsilon)}{(8+\epsilon)^2}} = n^{-\frac{8\epsilon-2\epsilon^2}{(8+\epsilon)^2}} = n^{-\Omega(1)}.$$

Thus, we can translate the weak PPRG G into a standard PPRG by Theorem 6.6.9, where we use the junta-composition condition of degree-d $\mathbb{F}_2$-polynomials. □

Corollary 6.8.3 (The third item of Corollary 6.2.6)**.** *For any $\epsilon \in (0,1)$, if d-LRPDT is $(n^{1+\epsilon}, n^{-(1+\epsilon)})$-hard on an example distribution samplable with shared randomness for some $d \in \mathbb{N}$, then a collection of PPRGs in $\oplus$-NC^0 exists.*

Proof. Let $r(n) = n^{1+\epsilon/4}$ and $p(n) = n^{1+\epsilon}$. Again, we use the observation that d-LRPDT is regarded as learning degree-d $\mathbb{F}_2$-polynomials selected by a degree-$(d+1)$ $\mathbb{F}_2$-polynomial by using an $r(n)$-bit random seed. Thus, we can apply the proof of Theorem 6.7.8 (without the XOR lemma) and convert the hardness of d-LRPDT with advantage $1/p(n)$ and sample complexity $r(n)^{1+\epsilon/4}$ into a collection of weak PPRGs that stretches an $r(n)$-bit random seed to an $r(n)^{1+\epsilon/4}$-bit pseudorandom string, and the indistinguishable parameter is $r(r^{-1}(n))^{1+\epsilon/4}/p(r^{-1}(n)) = n^{1+\epsilon/4}/p(r^{-1}(n))$, where we interpret the upper bound on the advantage $r(n)^{1+\epsilon/4}/p(n)$ of distinguishers as a function in the seed length $r(n)$. For sufficiently large $n \in \mathbb{N}$, we have $r(n)^{1+\epsilon/4} = n^{(1+\epsilon/4)^2} \leq n^{1+\epsilon/2+(\epsilon/2)^2} \leq n^{1+\epsilon}$.

Thus, the hardness assumption of learning satisfies the requirement on the sample complexity in Corollary 6.8.3. Furthermore, the indistinguishable parameter is at most

$$\frac{n^{1+\frac{\epsilon}{4}}}{p(r^{-1}(n))} = \frac{n^{1+\frac{\epsilon}{4}}}{n^{\frac{1+\epsilon}{1+\frac{\epsilon}{4}}}} = \frac{n^{1+\frac{\epsilon}{4}}}{n^{1+\frac{3\epsilon}{4+\epsilon}}} = n^{-\frac{8\epsilon-\epsilon^2}{4(4+\epsilon)}} = n^{-\Omega(1)}.$$

Thus, we can translate the collection of weak PPRGs into a collection of PPRGs by Theorem 6.6.9, where we use the junta-composition condition of degree-d $\mathbb{F}_2$-polynomials. □

6.9 Impossibility of Dualization of NC^0

In this section, we show that dualization of $O(1)$-junta (i.e., Boolean-valued functions in NC^0) is impossible even in the statistical setting. The formal statement is the following.

Theorem 6.9.1. *$O(1)$-junta is not dualizable, i.e., for every $k, k' \in \mathbb{N}$, there is no pair of functions g and h satisfying that for every $n \in \mathbb{N}$, $x \in \{0,1\}^n$, and every k-junta function $f\colon \{0,1\}^n \to \{0,1\}$,*

1. *$x^* := g(x)$ is a k'-junta function of input size $n' = \mathsf{poly}(n)$;*
2. *$f^* := h(f) \in \{0,1\}^{n'}$; and*
3. *$f(x) = x^*(f^*)$.*

To show Theorem 6.9.1, we recall the notion of 2-round communication protocols.

Definition 6.9.2 (2-round communication protocol)**.** *Let $\mathscr{C}$ be a concept class. A 2-round communication protocol for evaluating $\mathscr{C}$ is a pair of deterministic algorithms* $(\mathsf{Input}, \mathsf{Function})$ *(they are possibly not efficiently computable) satisfying that there exist functions $m_{\mathsf{input}}(n)$ and $m_{\mathsf{function}}(n)$ such that for every $n \in \mathbb{N}$, every $x \in \{0,1\}^n$, and every $f \in \mathscr{C}_n$,*

1. Input *takes x as input and sends a message $a \in \{0,1\}^{m_{\mathsf{Input}}(n)}$ to* $\mathsf{Function}$*.*
2. $\mathsf{Function}$ *takes f and the message a as input and sends a message $b \in \{0,1\}^{m_{\mathsf{function}}(n)}$ to* Input*.*
3. Input *obtains the message b additionally and outputs $f(x)$.*

For convenience, we call a 2-round communication protocol for evaluating $\mathscr{C}$ with the message-length functions $m_{\mathsf{input}}(n)$ and $m_{\mathsf{function}}(n)$ an $(m_{\mathsf{input}}(n), m_{\mathsf{function}}(n))$-protocol for evaluating $\mathscr{C}$.

Any concept class $\mathscr{C}$ has a trivial $(n,1)$-protocol (for evaluating $\mathscr{C}$), where Input sends $x \in \{0,1\}^n$, and $\mathsf{Function}$ sends back $f(x) \in \{0,1\}$. Thus, nontrivial cases are when Input does not send the whole input x.

Now, we show Theorem 6.9.1 by observing that any dualization of NC^0 yields a 2-round communication protocol for evaluating $O(1)$-junta with short messages, but such a protocol does not exist information theoretically.

Proof of Theorem 6.9.1. Fix $k, k' \in \mathbb{N}$ arbitrarily. Theorem 6.9.1 follows from Claims 6.9.3 and 6.9.4.

Claim 6.9.3. *If there exist the functions g and h for dualization as in Theorem 6.9.1, then there exists an $(O(\log n), O(1))$-protocol for evaluating k-junta.*

Claim 6.9.4. *For any $\epsilon > 0$, there is no $((1-\epsilon)n, o(\log n))$-protocol for evaluating k-junta.*

Proof of Claim 6.9.3. We can construct an $(O(\log n), O(1))$-protocol for evaluating k-junta based on g and h as follows: for any $n \in \mathbb{N}$, $x \in \{0,1\}^n$, and any k-junta function $f\colon \{0,1\}^n \to \{0,1\}$,

1. $\mathsf{Input}(x)$ computes the dual $x^* = g(x)$ and sends all indices $i_1, \dots, i_{k'} \in [n']$ of the relevant variables of x^* to $\mathsf{Function}$, where the message length is at most $O(k' \log n') = O(\log n)$.
2. $\mathsf{Function}(f; i_1, \dots, i_{k'})$ computes the dual $f^* \in \{0,1\}^{n'}$ and sends $f^*_{i_1}, \dots, f^*_{i_{k'}} \in \{0,1\}$ that are relevant to computing $x^*(f^*)$ to Input, where the message length is at most $O(k') = O(1)$.
3. Input, given $f^*_{i_1}, \dots, f^*_{i_{k'}}$, computes and outputs $x^*(f^*) = f(x)$.

$\diamond$

Proof of Claim 6.9.4. Suppose that there exists a $((1-\epsilon)n, m(n))$-protocol $(\mathsf{Input}, \mathsf{Function})$ for evaluating k-junta, where $\epsilon > 0$ and $m(n) = o(\log n)$. We derive a contradiction.

Fix a sufficiently large $n \in \mathbb{N}$ with $\epsilon n - 1 \geq 2^{m(n)} = o(n)$ arbitrarily. We can classify each input string $x \in \{0,1\}^n$ according to the massage sent by $\mathsf{Input}(x)$. Since the length of the message is $(1-\epsilon)n$, the number of possible messages is at most $2^{(1-\epsilon)n}$. Thus, there exists a message $a \in \{0,1\}^{(1-\epsilon)n}$ such that $S_a = \{x \in \{0,1\}^n : \mathsf{Input}(x) \text{ sends } a\}$ has cardinality at least $2^n / 2^{(1-\epsilon)n} = 2^{\epsilon n}$.

We focus on the case in which the given input x is contained in S_a. By the definition of S_a, the first message sent by Input is fixed to a. Thus, the second message sent by $\mathsf{Function}$ is determined only by a given k-junta function f. Again, we classify k-junta functions according to the second message as follows: for every $b \in \{0,1\}^{m(n)}$,

$$T_b = \{f\colon \{0,1\}^n \to \{0,1\} \mid f \text{ is } k\text{-junta and } \mathsf{Function}(f; a) \text{ sends } b\}.$$

We show that there exist $x \in S_a$ and $f, f' \in T_b$ such that $f(x) \neq f'(x)$ holds. This contradicts the correctness of $(\mathsf{Input}, \mathsf{Function})$ because, in the both cases of $\{(x, f), (x, f')\}$, the transcript is the same (i.e., the first message is a, and the second message is b); thus, Input cannot distinguish between f and f' and outputs the same value $y \in \{0,1\}$, and $f(x) = f'(x) = y$ must hold for the correctness. Thus, our goal is to find such $x \in S_a$ and $f, f' \in T_b$.

Let $d = |S_a| \geq 2^{\epsilon n}$. For each $j \in [n]$, we define $v^j \in \{0,1\}^d$ as $v^j = v^j_1 \cdots v^j_d$, where v^j_i is the j-th bit of the i-th string x in S_a (in lexicographic order) for each $i \in [d]$. If there are at most c distinct vectors in $v^1, \dots, v^n$ (say, $v^{j_1}, \dots, v^{j_c}$), then the cardinality of S_a is at most 2^c because each $x \in S_a$ is determined only by the patterns of $(v^{j_1}_i, \dots, v^{j_c}_i) \in \{0,1\}^c$, where i is the lexicographic order of x in S_a. Since $|S_a| \geq 2^{\epsilon n}$, there are at least $c \geq \epsilon n$ distinct vectors $v^{j_1}, \dots, v^{j_c}$ in $v^1, \dots, v^n$.

We consider 1-junta functions (i.e., k-junta functions) $\chi_{j_1}, \dots, \chi_{j_c}$, where $\chi_{j_\ell}(x) = x_{j_\ell}$ for each $\ell \in [c]$. Remember that the number of the separation $\{T_b\}_{b \in \{0,1\}^{m(n)}}$ of k-junta functions is at most $2^{m(n)} \leq \epsilon n - 1 \leq c - 1$. Thus, by the pigeonhole principle, there exist $\ell, \ell' \in [c]$ and $b \in \{0,1\}^{m(n)}$ such that $\chi_{j_\ell}, \chi_{j_{\ell'}} \in T_b$. Since v^{j_ℓ} and $v^{j_{\ell'}}$ are distinct vectors, there exists $i \in [n]$ such that $v^{j_\ell}_i \neq v^{j_{\ell'}}_i$. Let $x \in S_a$ be the i-th string in S_a.

We verify that $x \in S_a$ and $\chi_{j_\ell}, \chi_{j_{\ell'}} \in T_b$ satisfy the condition that $\chi_{j_\ell}(x) \neq \chi_{j_{\ell'}}(x)$ for contradiction as follows:

$$\chi_{j_\ell}(x) = x_{j_\ell} = v^{j_\ell}_i \neq v^{j_{\ell'}}_i = x_{j_{\ell'}} = \chi_{j_{\ell'}}(x).$$

$\diamond$

$\square$

Chapter 7

PACland: A World Where PAC Learning is Easy

In previous chapters, we investigated learning theoretic implications in Heuristica and Pessiland, which are the possible worlds involved in the core concepts (i.e., average-case hardness of NP and one-way functions) in computational complexity theory and cryptography. However, one natural and important question was not addressed there, i.e., what is the *exact* capability of the statement that "PAC learning all efficiently computable concepts is easy" particularly when there is no restriction on hypotheses? We name the possible world where PAC learning all efficiently computable concepts (particularly, P/poly) is easy as *PACland* for familiarity and investigate its complexity-theoretic and cryptographic aspects.

The concept class P/poly of polynomial-size circuits is complete in efficient PAC learning for efficiently computable concepts in the sense that (i) P/poly is efficiently computable; and (ii) if a class $\mathscr{C}$ is efficiently PAC learning, then $\mathscr{C} \subseteq \mathsf{P/poly}$ [Sch90]. Furthermore, by a simple padding argument, we can assume that the size is restricted to n^2 without loss of generality (see Proposition 7.1.3). Therefore, in this chapter, we mainly consider the PAC learnability of the class $\mathsf{SIZE}[n^2]$ to discuss PACland.

7.1 Additional Preliminaries

We introduce additional preliminaries required only in this chapter.

Universal Circuit and Circuit Translator

We fix a proper encoding for Boolean circuits. For any circuit C, we use $\langle C \rangle$ to explicitly denote the binary encoding of C. Otherwise, we may abuse the same notation C for the encoding.

For convenience, we assume the following: (1) every binary string $u \in \{0,1\}^*$ represents some circuit C_u, which is done by assigning invalid encodings u to the trivial circuit $C_u(x) = 0$; (2) zero-padding is available, specifically, the minimum valid encoding u for any circuit C ends with 1, and we regard $u \circ 0^*$ is also valid encoding of the circuit C. For any $s \in \mathbb{N}$, let $e(s) = O(s \log s)$ be the size of minimum encoding for s-size circuits satisfying the assumptions above.

The following language C-Eval is known as a P-complete problem.

$$\mathsf{C\text{-}Eval} = \{\langle C, x\rangle : C \text{ is an } n\text{-input circuit and } x \in \{0,1\}^n \text{ satisfying } C(x) = 1\}.$$

In addition, the class P is equivalent to the class recognized by uniformly generated polynomial-size circuits [cf. AB09]. These results lead to the following facts.

Lemma 7.1.1 (Universal circuits)**.** *There exists a polynomial-size circuit family $UC = \{UC_N\}_{N\in\mathbb{N}}$ satisfying that for any $u, x \in \{0,1\}^*$,*

$$UC_N(\langle u, x\rangle) = \begin{cases} C_u(x) & (\text{if } u \text{ represents an } n\text{-input circuit } C_u \text{ and } |x| = n) \\ 0 & (\text{otherwise}), \end{cases}$$

where $N = |\langle u, x\rangle|$. Moreover, UC is uniformly generated, that is, there exists a polynomial-time Turing machine UCM such that $\langle UC_N\rangle \leftarrow UCM(1^N)$ for any $N \in \mathbb{N}$.

Lemma 7.1.2 (Circuit translator)**.** *There exists a polynomial-time Turing machine CT such that for the input $\langle M, 1^n\rangle$ where M is a description of a $T(n)$-time Turing machine and $n \in \mathbb{N}$, CT outputs $\langle C\rangle \in \{0,1\}^{e(s(n))}$ where C is an n-input circuit of size $s(n) = O(T(n)\log T(n))$ satisfying $C(x) = M(x)$ for any $x \in \{0,1\}^n$. We call the above-mentioned CT a circuit translator.*

In the second argument of CT, we may use the notation $1^{n_1\times\cdots\times n_k}$ to denote the length of binary encoding for a k-tuple of binary strings $(x_1, \ldots, x_k)$ where $x_i \in \{0,1\}^{n_i}$ for each $i \in [k]$.

We also define a size-bounded universal circuit

$$UC^{s(n)} = \{UC_n^{s(n)} \colon \{0,1\}^{e(s(n))} \times \{0,1\}^n \to \{0,1\}\}_{n\in\mathbb{N}}$$

by

$$UC_n^{s(n)}(u, x) = \begin{cases} UC(\langle u, x\rangle) & \text{if } u = \langle C\rangle \text{ and } C \in \mathsf{SIZE}[s(n)] \\ 0 & \text{otherwise.} \end{cases}$$

Note that, by the standard construction of universal circuits [cf. AB09],

$$(\text{the size of } UC_n^{s(n)}) = \tilde{O}(s(n) + (s(n) - n)^2).$$

In addition, $\mathsf{SIZE}[s(n)]$ is evaluated by $UC^{s(n)}$ as

$$\mathsf{SIZE}[s(n)]_n = \left\{C(x) := UC_n^{s(n)}(u, x) : u \in \{0,1\}^{e(s(n))}\right\} \text{ for each } n \in \mathbb{N}.$$

Therefore, we fix $UC^{s(n)}$ for the evaluation function for $\mathsf{SIZE}[s(n)]$, which determines the dual $\mathsf{SIZE}[s(n)]^*$ of $\mathsf{SIZE}[s(n)]$.

Padding Argument

Proposition 7.1.3. *For any polynomial $s(n) > n$, $\mathsf{SIZE}[s(n)]$ is PAC learnable iff $\mathsf{SIZE}[n^2]$ is PAC learnable.*

Proof. We only need to consider the case in which $s(n) \geq n^2$. We construct a PAC learner L_s for $\mathsf{SIZE}[s(n)]$ from another PAC learner L for $\mathsf{SIZE}[n^2]$. Suppose that $L(1^n, 1^{\epsilon^{-1}}, 1^{\delta^{-1}})$ needs $m(n, \epsilon, \delta)$ samples for learning, where m is a polynomial.

Let n be the size of examples for L_s. First, $L_s(1^n, 1^{\epsilon^{-1}}, 1^{\delta^{-1}})$ calculates the minimum value $n' \in \mathbb{N}$ satisfying $s(n) - n \leq n'^2 - n'$. Obviously, n' is polynomially related to n. Second, L_s takes $m(n', \epsilon, \delta)$ examples and executes $L(1^{n'}, 1^{\epsilon^{-1}}, 1^{\delta^{-1}})$, where L_s pads each example $x \in \{0,1\}^n$ with

0 as $x \circ 0^{n'-n} \in \{0,1\}^{n'}$. Finally, if L returns a hypothesis $h\colon \{0,1\}^{n'} \to \{0,1\}$, then L_s outputs a hypothesis $h'\colon \{0,1\}^n \to \{0,1\}$ defined as $h'(x) = h(x \circ 0^{n'-n})$.

Let C be a circuit of size $s(n)$ which computes the target function L_s tries to learn, and let $\mathcal{D}$ be an example distribution on $\{0,1\}^n$. Then it is easily verified that L_s executes L with a valid setting where the target function is computed by C (discarding a suffix of length $n'-n$) and the example distribution $\mathcal{D} \circ 0^{n'-n}$. Although the target circuit for L needs additional $n'-n$ input-gates, the size is bounded above by $s(n)+n'-n \leq n'^2$. Therefore, L succeeds in learning with the accuracy parameter ϵ and the confidence parameter δ, and so does L_s. □

7.2 Auxiliary-Input Primitive Corresponding to Hardness of PAC Learning

First, we consider which type of cryptographic primitives corresponds to the hardness of PAC learning. Our approach is to apply the idea outlined in Section 5.2.1 to obtain the cryptographic characterization of the hardness of PAC learning. As a result, we characterize the hardness of PAC learning by the existence of the auxiliary-input variant of a hitting-set generator (HSG) with an additional locality condition.

HSG is a variant of PRG that has a weaker (one-sided-error) security condition, which was originally introduced by Andreev, Clementi, and Rolim [ACR98] to develop a general derandomization method (see also [GVW11; Gol11b]). For the original purpose of the derandomization, we require HSG in computational complexity regime, i.e., an exponential-stretch HSG secure against super-polynomial-time adversaries. By contrast, we consider an HSG in cryptographic regime, i.e., a polynomial-stretch HSG secure against polynomial-time adversaries, which was also get an attention independently in context of meta-complexity [San20].

To characterize the PAC learnability, we first weaken the condition of HSG by introducing the auxiliary-input variant. At this point, the existence of the *auxiliary-input* HSG becomes weaker than the hardness of PAC learning. It is worthy of note that this is the first (weakened) pseudorandom generator whose existence is weaker than the hardness of PAC learning, as far as we know.

Definition 7.2.1 (Auxiliary-input hitting set generator)**.** *A polynomial-time-computable auxiliary-input function $G = \{G_z\colon \{0,1\}^{n(|z|)} \to \{0,1\}^{\ell(n(|z|))}\}_{z\in\{0,1\}^*}$ is an auxiliary-input hitting set generator (AIHSG) if $\ell(n) > n$ and for any polynomial-time probabilistic algorithms A, there exists a non-empty set $Z_A \subseteq \{0,1\}^*$ such that for every $z \in Z_A$, G_z hits the language recognized by of A, that is, A satisfies either*

$$\exists x \in \{0,1\}^{|z|} \text{ s.t. } \Pr_A[A(z, G_z(x)) = 1] \geq 2/3 \quad \text{or} \quad \Pr_{y\sim\{0,1\}^{\ell(n(|z|))}}\left[\Pr_A[A(z,y)=1] \geq 2/3\right] \leq 1/2.$$

We call the function $\ell(\cdot)$ in $n(|z|)$ the stretch of G.

Indeed, it is not so difficult to see that if we regard auxiliary-input of AIHSG as input, then the resulting generator becomes a secure HSG. However, the stretch (i.e., how long the hidden input is stretched) decreases drastically because the input length is stretched from $n(|z|)$ to $|z| + n(|z|)$.

Next, we add two conditions to AIHSG. Then the existence of such modified HSGs will exactly coincide with the hardness of PAC learning. The first condition is arbitrary polynomial stretch. In

the case of PRGs, we can easily construct any polynomial stretch PRG from any PRG even if the original stretch is $n+1$. In the case of AIHSGs, however, such construction is quite non-trivial. The second condition is *locality*, i.e., that condition that each bit of the output is locally computable from the input and a small part of auxiliary-input, and the way to compute does not depend on the values of input and auxiliary-input. To sum up, the modified AIHSG is formulated as follows.

Definition 7.2.2 (Local AIHSG)**.** *A polynomial-time computable auxiliary-input function $G = \{G^\tau\}_{\tau\in\mathbb{N}}$ is a local AIHSG if G satisfies the following two conditions:*

- (*Arbitrary polynomial stretch*) *for any τ, $G^\tau = \{G^\tau_z\}_{z\in\{0,1\}^*}$ is an AIHSG with the stretch $n(|z|)^\tau$;*
- (*Locally computable*) *there exists a non-adaptive polynomial-time oracle machine $\mathrm{local}G^?$ such that $\mathrm{local}G^z(\tau, |z|, i, x)$ returns the i-th bit of $G^\tau_z(x)$ and the positions of queries determined by $\tau, |z|, i$ (not depending on input x and auxiliary-input z).*

Then, the existence of a local AIHSG *exactly* characterizes the hardness of the standard PAC learning.

Theorem 7.2.3 (Hardness of PAC learning $\Leftrightarrow$ local AIHSG)**.** $\mathsf{P/poly}$ (*particularly* $\mathsf{SIZE}[n^2]$) *is not PAC learnable if and only if there exists a local AIHSG.*

The reader may ask a candidate for local AIHSG. We also give the explicit generator which is complete in the sense that it must be local AIHSG if there exists some local AIHSG. Namely, the explicit generator must be local AIHSG if PAC learning $\mathsf{SIZE}[n^2]$ is hard.

Theorem 7.2.4 (Complete local AIHSG)**.** *There exists an explicit generator $\{UC^\tau\}_{\tau\in\mathbb{N}}$ satisfying that $\{UC^\tau\}_{\tau\in\mathbb{N}}$ is a local AIHSG iff there exists a local AIHSG.*

Intuition: Duality between PAC Learning and Local AIHSG

We give an intuition of the correspondence between the hardness of PAC learning and local AIHSG. The argument mainly follows the ideas in Section 5.2.1; thus, we strongly recommend the reader to refer to Section 5.2.1 first.

Remember that we applied the Occam's algorithm (i.e., finding a consistent hypothesis) to see the correspondence between the hardness of PAC learning (in the BFKL model) and AIOWF. In this section, we apply another formulation of PAC learning, i.e., RRHS-refutation introduced by Vadhan [Vad17]. We remark that the task of RRHS-refutation for a concept class $\mathscr{C}$ is, for a given sample set $(x_1, b_1), \ldots, (x_m, b_m)$, to distinguish between (i) the "realizable" case in which there exists a function $f \in \mathscr{C}$ such that $f(x_i) = b_i$ for any i and (ii) the "random" case in which the labels $b_1, \ldots, b_m$ are selected uniformly at random. Particularly, we consider the case in which $\mathscr{C} = \mathsf{SIZE}[n^2]$.

To see the correspondence between the hardness of PAC learning and local AIHSG, we regard the examples $x_1, \ldots, x_m$ as auxiliary-input and the hidden function f as the random seed for AIHSG. Then, it is not hard to verify that the task of RRHS-refutation exactly corresponds to the task of distinguishing pseudorandom strings in the image of AIHSG from random strings. Technically, as seen in Section 6.4.1, this argument only yields the correspondence between a fixed sample complexity and a fixed stretch of a generator. To extend this to the case of PAC learnability with arbitrarily large polynomial sample complexity, we use the polynomial-stretch condition of local

AIHSG. In addition, we require the local condition to preserve the property of learning that each bit is computed locally by a target function, even in a generator. Without the locality condition, in the implication from the existence of AIHSG to the hardness of PAC learning, there is no guarantee on the upper bound on the complexity of computing each label (i.e., pseudorandom bit) because some output bit can heavily depend on computations of other output bits in general AIHSGs.

7.2.1 Proof of Theorems 7.2.3 and 7.2.4

We again use the notion of *dualization* (especially for $\mathsf{P/poly}$). Let $E = \{E_n\}_{n\in\mathbb{N}}$ be a family of function $E_n\colon \{0,1\}^{s(n)} \times \{0,1\}^n \to \{0,1\}$. We say that a class $\mathscr{C} = \{\mathscr{C}_n\}_{n\in\mathbb{N}}$ is evaluated by E if $\mathscr{C}_n = \{f(x) := E_n(u,x) : u \in \{0,1\}^{s(n)}\}$ for any $n \in \mathbb{N}$. Remember that we define the dual-class $\mathscr{C}^* = \{\mathscr{C}^*_n\}_{n\in\mathbb{N}}$ as $\mathscr{C}^*_n = \{g(u) := E_n(u,x) : x \in \{0,1\}^n\}$. Now, we introduce the following lemma, which was first used by [Vad17].

Lemma 7.2.5 ([PW90; Vad17])**.** *Let $\mathscr{C}$ and $\mathscr{C}'$ be concept classes evaluated by $\{E_n\colon \{0,1\}^{s(n)} \times \{0,1\}^n \to \{0,1\}\}_{n\in\mathbb{N}}$ and $\{E'_n\colon \{0,1\}^{t(n)} \times \{0,1\}^n \to \{0,1\}\}_{n\in\mathbb{N}}$, respectively. Suppose that for any $n \in \mathbb{N}$, there exist $n' \in \mathbb{N}$, and polynomial-time computable functions $f_{rep}\colon \{0,1\}^n \to \{0,1\}^{t(n')}$ and $f_{ex}\colon \{0,1\}^{s(n)} \to \{0,1\}^{n'}$ satisfying that*

$$E'_{n'}(f_{rep}(x), f_{ex}(u)) = E_n(u,x).$$

Then the following holds:

- *if $\mathscr{C}'$ is RRHS-refutable, then $\mathscr{C}^*$ is also RRHS-refutable;*
- *if $\mathscr{C}'^*$ is RRHS-refutable, then $\mathscr{C}$ is also RRHS-refutable.*

We apply Lemma 7.2.5 to polynomial-size circuits. For any polynomial s, we use the notation $s^\dagger$ to denote the polynomial such that the size of UC^s_n is at most $s^\dagger(e(s(n)))$ for any $n \in \mathbb{N}$. Then we have the following lemma.

Lemma 7.2.6. *For any polynomial $s(n)$,*

- $\mathsf{SIZE}[s(n)]^*$ *is RRHS-refutable if* $\mathsf{SIZE}[s^\dagger(n)]$ *is RRHS-refutable;*
- $\mathsf{SIZE}[s(n)]$ *is RRHS-refutable if* $\mathsf{SIZE}[s^\dagger(n)]^*$ *is RRHS-refutable.*

Proof. We give functions $f_{ex}\colon \{0,1\}^{e(s(n))} \to \{0,1\}^{e(s(n))}$ and $f_{rep}\colon \{0,1\}^n \to \{0,1\}^{e(s^\dagger(e(s(n))))}$ in Lemma 7.2.5 by

$$f_{ex}(u) = u, \quad \text{and} \quad f_{rep}(x) = \langle C_x \rangle, \text{ where } C_x(u) = UC^{s(n)}_n(u,x).$$

Note that the input size of C_x is $e(s(n))$ and the size is at most $s^\dagger(e(s(n)))$ by the definition of $s^\dagger$. Since $UC^{s(n)}$ is generated in polynomial time, f_{ex} and f_{rep} are polynomial-time computable. In addition, these functions satisfy the conditions in Lemma 7.2.5 as follows:

$$UC^{s^\dagger(n)}_{e(s(n))}(f_{rep}(x), f_{ex}(u)) = UC^{s^\dagger(n)}_{e(s(n))}(\langle C_x \rangle, u) = C_x(u) = UC^{s(n)}_n(u,x).$$

□

First, we introduce candidates for the complete local AIHSG in Theorem 7.2.4, which is simply composed of concatenated universal circuits. Remember that $e(s) = O(s \log s)$ is the size of the binary encoding for s-size circuits satisfying the assumptions in Section 7.1.

Definition 7.2.7. *For any polynomial $s(n)$ and $\ell(n)$, we define a polynomial-time computable auxiliary-input functon $UC^{s,\ell} = \{UC^{s,\ell}_z \colon \{0,1\}^{n(|z|)} \to \{0,1\}^{\ell(n(|z|))}\}_{z\in\{0,1\}^*}$, where $n(|z|) = \max n \in \mathbb{N} : e(s(n)) \cdot \ell(n) \leq |z|$, as*

$$UC^{s,\ell}_z(x) = UC^s(z_1, x) \circ \cdots \circ UC^s(z_{\ell(n)}, x),$$

where $z_i \in \{0,1\}^{e(s(n))}$ for each $i \in [\ell(n)]$, $z = z_1 \circ \cdots \circ z_{\ell(n)} \circ z_{left}$, and $z_{left} \in \{0,1\}^{|z|-e(s(n))\cdot\ell(n)}$.

Now we show the following main theorem. Note that Theorem 7.2.3 follows from Item 1 $\Leftrightarrow$ Item 2.

Theorem 7.2.8. *The following are equivalent:*

1. $\mathsf{SIZE}[n^2]$ *is not PAC learnable.*
2. *There exists a local AIHSG.*
3. *There exists a polynomial $s(n)$ such that $\{UC^{s(n),n^\tau}\}_{\tau\in\mathbb{N}}$ is a local AIHSG.*

Proof. (1 $\Rightarrow$ 2) For any $\tau \in \mathbb{N}$, we construct an AIHSG $G^\tau = \{G^\tau_z\}_{z\in\mathbb{N}}$ for each $\tau \in \mathbb{N}$. Define functions m and n as

$$m := m(|z|) = \max m \in \mathbb{N} : e(m^2)^\tau \cdot m \leq |z|, \text{ and } n := n(|z|) = e(m^2).$$

Then m and n are polynomially related to $|z|$ and any binary string $z \in \{0,1\}^*$ is divided as

$$z = z_1 \circ \cdots \circ z_{e(m^2)^\tau} \circ z_{last},$$

where $|z_i| = m$ for each $i \in [e(m^2)^\tau]$ and $|z_{last}| = |z| - e(m^2)^\tau \cdot m$.

Now we define $G^\tau_z \colon \{0,1\}^n \to \{0,1\}^{n^\tau}$ as

$$G^\tau_z(x) = UC^{n^2}_m(x, z_1) \circ \ldots UC^{n^2}_m(x, z_{e(m^2)^\tau}).$$

The generator above satisfies the conditions in Definition 7.2.1 because we can calculate the i-th bit of $G^\tau_z(x)$ in polynomial-time in n by executing $UC^{n^2}_m(x, z_i)$, and the location of z_i in z is determined only by $\tau, |z|$ and i.

Suppose for contradiction that there exists τ such that G^τ is not an AIHSG, and let A be the adversary. By Theorem 2.3.7, it is enough to construct a refuter for $\mathsf{SIZE}[n^2]$ from A.

Now we define the refuter R for $\mathsf{SIZE}[n^2]$ with $e(n^2)^\tau$ samples as follows: $R(x_1, \ldots, x_{n^\tau}, b_1, \ldots, b_{n^\tau})$ executes $A(x_1 \circ \cdots \circ x_{e(n^2)^\tau}, b_1 \circ \cdots \circ b_{e(n^2)^\tau})$ and if A outputs 1 (resp. 0), then R outputs "random" (resp. "realizable").

It is easy to verify that R outputs "random" with probability at least $2/3$ for greater than half of random labels. For "realizable" cases, we assume that there exists $y \in \{0,1\}^{e(n^2)}$ such that $UC^{n^2}_n(y, x_i) = b_i$ for any $i \in [e(n^2)^\tau]$. Let $z = x_1 \circ \cdots \circ x_{e(n^2)^\tau}$. Then $m(|z|) = n$ and $n(|z|) = e(n^2)$, and we have

$$G^\tau_z(y) = UC^{n^2}_n(y, x_1) \circ \cdots \circ UC^{n^2}_n(y, x_{e(n^2)^\tau}) = b_1 \circ \cdots \circ b_{e(n^2)^\tau}.$$

Since $b_1 \circ \cdots \circ b_{e(n^2)^\tau}$ is contained in the image of G^τ_z, the adversary A outputs 0 with probability at least $2/3$. In this case, R outputs "realizable" with probability at least $2/3$.

($2 \Rightarrow 3$) Let $\{G^\tau\}_{\tau\in\mathbb{N}}$ be a local AIHSG and $localG^?$ be the polynomial-time oracle machine in the condition of Definition 7.2.2. Since the length of input $(\tau, |z|, i, x)$ is at most $\log\tau + \log|z| + \tau\log(n(|z|)) + n(|z|) \leq \mathsf{poly}(n(|z|))$, $localG$ halts in polynomial time in $n := n(|z|)$. Thus, we can assume that $localG^?$ queries $q(n)$ points for some polynomial q. Since the queries are non-adaptive and determined by only $\tau, |z|, i$, we can divide $localG^?$ into two Turing machines Pos and M satisfying that for any (sufficiently long) string z,

$$Pos(\tau, |z|, i) = (j_1, \ldots, j_{q(n)});$$
$$M(z_{j_1}, \ldots, z_{j_{q(n)}}, \tau, |z|, i, x) = localG^z(\tau, |z|, i, x) = G^\tau_{z,i}(x),$$

where Pos and M halt in $t(n)$-time for some polynomial t.

Now we show that $\{UC^{t(n)^2, n^\tau}\}_{\tau\in\mathbb{N}}$ is a local AIHSG. The conditions about locality are easily verified as in the proof of ($1 \Rightarrow 2$). For contradiction, we assume that there exists τ such that $UC^{t(n)^2,n^\tau}$ is not an AIHSG by the adversary A, and construct the adversary A' which breaks the security of G^τ by using A.

Algorithm 1: A'

Input : $z \in \{0,1\}^*$ and $y \in \{0,1\}^{n(|z|)^\tau}$

1 execute $C(,,,,) \leftarrow CT(M, 1^{q(n)\times\log\tau\times\log|z|\times\tau\log(n(|z|))\times n})$;
2 **for** $i := 1$ *to* n^τ **do**
3 execute $(j^i_1, \ldots, j^i_{q(n)}) \leftarrow Pos(\tau, |z|, i)$;
4 hardwire the indices as $C_i(x) = C(z_{j^i_1} \circ \ldots \circ z_{j^i_{q(n)}}, \tau, |z|, i, x)$, and $u_i = \langle C_i(x)\rangle$;
5 truncate/zero-pad u_i to the length $e(t(n)^2)$;
6 output the same value to $A(u_1 \circ \cdots \circ u_{n^\tau}, y)$;

We analyze the adversary A'. For sufficiently large $|z|$, the size of C in line 1 is at most $t(n(|z|))^2$. In this case, each $C_i(x)$ is properly encoded to the length $e(t(n)^2)$ in line 5.

For any auxiliary-input z to G^τ and $i \in [n^\tau]$, let u_i be the binary encoding given in line 5 and $u_z := u_1 \circ \cdots \circ u_{n^\tau}$ be auxiliary-input in line 6. Then we have that

$$\Pr_y\left[\Pr_{A'}[A'(z,y) = 1] \geq 2/3\right] = \Pr_y\left[\Pr_A[A(u_z, y) = 1] \geq 2/3\right] \geq 1/2.$$

If there exists $x \in \{0,1\}^n$ satisfying that $G^\tau_z(x) = y$, then

$$\begin{aligned} y = G^\tau_z(x) &= M(z_{j^1_1}, \ldots, z_{j^1_{q(n)}}, \tau, |z|, 1, x) \circ \cdots \circ M(z_{j^{n^\tau}_1}, \ldots, z_{j^{n^\tau}_{q(n)}}, \tau, |z|, n^\tau, x) \\ &= C_1(x) \circ \cdots \circ C_{n^\tau}(x) \\ &= UC^{t(n)^2}_n(u_1, x) \circ \cdots \circ UC^{t(n)^2}_n(u_{n^\tau}, x) \\ &= UC^{t(n)^2, n^\tau}_{u_z}(x). \end{aligned}$$

Thus, y is also contained in the image of $UC_{u_z}^{t(n)^2,n^\tau}$. Therefore, the adversary A outputs 0 with probability at least $2/3$, and so does A'. By the observations above, A' breaks the security of G^τ, which contradicts the assumption that $\{G^\tau\}_{\tau\in\mathbb{N}}$ is a local AIHSG.

(3 $\Rightarrow$ 1) For a polynomial $s(n)$, suppose that $\{UC^{s(n),n^\tau}\}_{\tau\in\mathbb{N}}$ is a local AIHSG. First, we show that $\mathsf{SIZE}[n^2]^*$ is not RRHS-refutable.

For contradiction, we assume that there exists a refuter R for $\mathsf{SIZE}[s(n)]^*$ with n^c samples. Now we construct an adversary A for $UC^{s(n),n^c}$ by

$$A(z,y) = \mathbb{1}\{R(z_1,\dots,z_{n^c},y_1,\dots,y_{n^c}) = \text{“random”}\}$$

where $z = z_1 \circ \cdots \circ z_{n^\tau} \circ z_{last}$, $|z_i| = e(n(|z|)^2)$, and $|z_{last}| = |z| - n^c \cdot e(n^2)$.

For any z, if y is randomly selected, then we have

$$\Pr_y\left[\Pr_A[A(z,y)=1] \geq 2/3\right] = \Pr_y\left[\Pr_R[R(z_1,\dots,z_{n^c},y_1,\dots,y_{n^c}) = \text{“random”}] \geq 2/3\right] \geq 1/2.$$

On the other hand, if there exists $x \in \{0,1\}^n$ such that $UC_z^{s(n),n^\tau}(x) = y$, then for each $i \in [n^c]$, $y_i = UC^{s(n)}(z_i, x)$. Therefore, the given sample is a “realizable” case for $\mathsf{SIZE}[s(n)]^*$, and A outputs 1 with probability at least $2/3$ by the correctness of R. Thus, A breaks the security of $UC^{s(n),n^c}$.

Now we have that $\mathsf{SIZE}[s(n)]^*$ is not RRHS-refutable. By taking a sufficiently large polynomial $t(n)$ such that $s(n) \leq t^\dagger(n)$, we conclude that

$$\begin{aligned}
&\mathsf{SIZE}[s(n)]^* \text{ is not RRHS-refutable} \\
&\quad\Longrightarrow \mathsf{SIZE}[t^\dagger(n)]^* \text{ is not RRHS-refutable} && (\because \mathsf{SIZE}[s(n)]^* \subseteq \mathsf{SIZE}[t^\dagger(n)]^*) \\
&\quad\Longrightarrow \mathsf{SIZE}[t(n)] \text{ is not RRHS-refutable} && (\because \text{Lemma 7.2.6}) \\
&\quad\Longrightarrow \mathsf{SIZE}[t(n)] \text{ is not PAC learnable.} && (\because \text{Theorem 2.3.7})
\end{aligned}$$

By Proposition 7.1.3, $\mathsf{SIZE}[n^2]$ is not PAC learnable. □

To show Theorem 7.2.4, the implication 2 $\Rightarrow$ 3 in Theorem 7.2.8 is not sufficient because the function $s(n)$ in the statement 3 depends on the size of local AIHSGs in the statement 2. However, this problem is easily resolved by applying Theorem 7.2.8 twice. The existence of local AIHSG implies hardness of PAC learning $\mathsf{SIZE}[n^2]$ (2 $\Rightarrow$ 1), and the hardness implies the existence of specific generator $\{UC^{s(n),n^\tau}\}_{\tau\in\mathbb{N}}$ (1$\Rightarrow$ 3). In this case, the function $s(n)$ is determined depending on the size of concept class, that is n^2, thus we can fix $s(n)$ beforehand.

7.3 Meta-PAC Learning is as Hard as PAC Learning

Next, we study a decision problem whose worst-case hardness exactly corresponds to the hardness of PAC learning. Previously, Blumer, Ehrenfeucht, Haussler, and Warmuth [BEHW87] and Schapire [Sch90] characterized PAC learning by the NP-search problem of finding a consistent hypothesis. However, since the latter problem is not known to be NP-complete, it is unclear whether the feasibility of general PAC learning for $\mathsf{P/poly}$ can be characterized by the computational complexity of a *decision* problem. Particularly, the characterization of PAC learning by RRHS-refutation (Theorem 2.3.7) seems to show that PAC learnability is a notion that corresponds to a variant of a decision problem that has a weaker requirement than determining the answer exactly.

In this section, we present a characterization of PAC learnability by the computational hardness of a decision problem. The problem we consider is involved with a natural question in computational learning theory: why is proving efficient learnability difficult? First, we hypothesize about this question and formulate it as a decision problem.

Our observation is the following: to prove learnability for a certain concept class, we must determine at least whether the concept class is polynomially learnable or not. We hypothesize that such a task of determining learnable classes and not-learnable classes is indeed computationally difficult. In the following, we focus on how to discuss the validity of this hypothesis.

Our idea is to formulate determining the polynomial learnability of a given concept class as a computational problem and analyze the computational complexity. However, we cannot get a straightforward formulation for such a computational task due to the following reasons. First, a concept class does not have a reasonable binary representation in general. Second, more crucially, polynomial learnability concerns with the asymptotic behavior when sufficiently abundant polynomial-size resources (i.e., samples and running-time) are available. Thus, we need to reflect the notion of "sufficiently abundant but polynomial-size resources" in our formulation.

To resolve the first problem, we focus on an evaluation function for a concept class. Usually, a concept class which we are interested in has an efficiently computable evaluation rule: for example, from input x and a simple disjunctive normal form formula ϕ, we can easily evaluate the value of $\phi(x)$. Therefore, we regard a polynomial-size evaluation circuit for a concept class as the definition of the concept class. Since Boolean circuits have binary encoding, this formulation enables us to encode concept classes and to define a language composed of concept classes.

We define a language $\mathsf{Learnable}$ representing efficiently learnable classes as follows. Note that the third condition in the following definition captures the notion of "PAC learnable", which is motivated by the formulation of PAC learnability by RRHS-refutation [Vad17] (where we consider non-uniform learners).

Definition 7.3.1 ($\mathsf{Learnable}_{\tau,m}$)**.** *For any polynomial $\tau := \tau(n)$ and $m := m(n)$, we define a language $\mathsf{Learnable}_{\tau,m}$ as a collection of circuits C satisfying that:*

- *C computes an evaluation function $E\colon \{0,1\}^{s(n)} \times \{0,1\}^n \to \{0,1\}$ for some $n \in \mathbb{N}$,*
- *the size of C is at most $\tau(n)$,*
- *there exists a circuit[1] $L \in \mathsf{SIZE}[n^2]$ which learns the concept class determined by C with m samples in the following sense: for any $x_1, \ldots, x_m \in \{0,1\}^n$, L satisfies that*
  - *For any $u \in \{0,1\}^{s(n)}$, $L(x_1, \ldots, x_m, E(u, x_1), \ldots, E(u, x_m)) = 0$;*
  - $\Pr_{b_1,\ldots,b_m \sim \{0,1\}}[L(x_1, \ldots, x_m, b_1, \ldots, b_m) = 1] \geq 1/2$.

Note that the third condition in Definition 7.3.1 captures the notion of "PAC learnable", which is motivated by the formulation of PAC learnability by RRHS-refutation [Vad17] (where we consider non-uniform learners). Formally, we can verify that the language $\mathsf{Learnable}_{\tau,m}$ represents PAC learnability in the following sense.

Theorem 7.3.2. *Let $\mathscr{C}$ be a concept class whose evaluation function is computed by a circuit $C_n\colon \{0,1\}^{s(n)} \times \{0,1\}^n \to \{0,1\}$ of size $\tau(n)$, that is, $\mathscr{C}_n = \{f(x) := C_n(u, x) : u \in \{0,1\}^{s(n)}\}$. The class $\mathscr{C}$ is non-uniformly PAC learnable iff for sufficiently large c, $C_n \in \mathsf{Learnable}_{\tau,n^c}$ for each n.*

[1] In fact, any exponent factor of the bound on the size of L will not change the asymptotic complexity of $\mathsf{Learnable}$ (formally, the proof of Theorem 7.3.2 in Section 7.3.1).

In addition, $\mathsf{Learnable}_{\tau,m}$ is contained in the polynomial hierarchy. This is immediately derived from the definition of $\mathsf{Learnable}_{\tau,m}$ and the well-known result that $\mathsf{BPP} \subseteq \Sigma_2^p \cap \Pi_2^p$ [Lau83]. We prove Theorems 7.3.2 and 7.3.3 in Section 7.3.1.

Theorem 7.3.3. *For any polynomials τ and m, $\mathsf{Learnable}_{\tau,m} \in \Sigma_3^p$. Moreover, if $\mathsf{P} = \mathsf{BPP}$, then indeed $\mathsf{Learnable}_{\tau,m} \in \Sigma_2^p$.*

By Theorem 7.3.2, if we can efficiently recognize $\mathsf{Learnable}_{\tau,n^c}$ for sufficiently large c, then we can also efficiently determine the learnability of concept classes evaluated by $\tau(n)$-size circuits. In the opposite direction, if we can efficiently determine the learnability of such concept classes, then we can also efficiently recognize $\mathsf{Learnable}_{\tau,n^c}$ for sufficiently large c because $\mathsf{Learnable}_{\tau,n^c}$ corresponds to the set of learnable classes for sufficiently large c. Therefore, determining the learnability of concept classes evaluated by $\tau(n)$-size circuits corresponds to recognizing $\mathsf{Learnable}_{\tau,n^c}$ for sufficiently large c. This observation is central to resolve the second problem for the formulation. In addition, for sufficiently large c, the polynomial $\tau(n)$ does not affect the asymptotic complexity (formally, Theorem 7.3.10). Therefore, without loss of generality, we can fix τ as $\tau(n) = n^2$. These arguments lead to the following formulation of the hardness of determining polynomial PAC learnability. We name the task meta-PAC learning.

Definition 7.3.4 (Meta-PAC learning is hard)**.** *We say that meta-PAC learning is hard if there exists infinitely many $c \in \mathbb{N}$ satisfying that $\mathsf{Learnable}_{n^2,n^c} \notin \mathsf{P}$.*

The third result is the equivalence between the hardness of meta-PAC learning and the hardness of non-uniform PAC learning for polynomial-sized circuits. This result insists that a hard-to-learn class itself yields the hardness of determining which concept classes are indeed hard-to-learn.

Theorem 7.3.5 (Meta-PAC learning is as hard as learning)**.** *Meta-PAC learning is hard if and only if polynomial-size circuits are not non-uniformly PAC learnable.*

To the best of our knowledge, our meta-PAC learning problem is the first formulation of determining polynomial PAC learnability. There are several related work which has the same purpose of determining some notions related to learnability: e.g., deciding VC dimension [Sch99], and the several consistency problems proposed by [KT91]. However, their formulations do not capture the notion of polynomially PAC learnable. Indeed, the complexity of their problems is exactly characterized by the polynomial hierarchy (e.g., deciding VC dimension is Σ_3^p-complete, and the consistency problems are Σ_2^p-complete). These facts show the differences from our meta-PAC learning because PAC learnability is not characterized by such classes at present. A related problem in computational complexity is the minimum circuit-size problem (MCSP) [KC00]. MCSP informally asks whether the given language, equivalently a truth-table of a boolean function, is recognized by small size circuits. There are several differences from our meta-PAC learning, which arise from the difference between settings of computing and learning; for example, the term "efficient" represents exponentially different running-time because a truth-table of Boolean function has exponential-size representation in general.

Theorem 7.3.5 shows the possibility that a hard-to-learn class itself can be a cause of the hardness of proving efficient learnability. Although our proof is quite simple and follows from only fundamental knowledge in complexity theory (see below), to the best of our knowledge, no one has not insisted on this possibility. In this sense, the notion of hard-to-learn can be much harder to handle than we expected.

Proof Sketch of Theorem 7.3.5

We give an outline of proof of Theorem 7.3.5, which is very simple and intuitive. For simplicity, we omit the argument about the upper bound $\tau(n)$ on the size of evaluation circuits. Note that we consider the non-uniform model (i.e., circuits) as a standard computational model for learners.

From PAC Learning to Meta-PAC Learning. This direction is trivial from Theorem 7.3.2. Suppose that all polynomial-size circuits are PAC learnable. Notice that if a concept class is evaluated by a $\tau(n)$-size circuit, then the class consists only of $\tau(n)$-size circuits. Therefore, for sufficiently large c, $\mathsf{Learnable}_{\tau(n),n^c}$ exactly corresponds to the set of $\tau(n)$-size circuits. We can easily verify whether the size of a given circuit is at most $\tau(n)$ by a polynomial-time Turing machine.

From Meta-PAC Learning to PAC Learning. We show that if PAC learning polynomial-size circuits is hard, then meta-PAC learning is also hard. Suppose for contradiction that PAC learning polynomial-size circuits is hard, but meta-PAC learning is not hard.

By the assumption, there exist a polynomial-time Turing machine (a meta-PAC learner) determining PAC learnability and a circuit C evaluating a hard-to-learn concept class. Then we can construct a polynomial-time Turing machine A for solving the circuit-SAT problem. For any given circuit C', the algorithm A constructs a new concept class $\mathscr{C}$ evaluated by $C''(u_1 \circ u_2, x) := C(u_1, x) \wedge C'(u_2)$ and feeds C'' to the meta-PAC learner. Note that the input to C' is regarded as a part of the representation of target functions.

If the given C' is unsatisfiable, then the concept class $\mathscr{C}$ consists only of the constant function 0, which is trivially PAC learnable. On the other hand, if the given C' is satisfiable, then $\mathscr{C}$ contains the original concept class evaluated by C. Therefore, $\mathscr{C}$ must be hard-to-learn, and the meta-PAC learner can distinguish these two cases in polynomial-time. Thus, A can solve the circuit-SAT problem efficiently, which yields $\mathsf{P} = \mathsf{NP}$.

As seen in Chapter 2.6, if $\mathsf{P} = \mathsf{NP}$, then we can construct an efficient PAC learning algorithm for polynomial-size circuits [BEHW87]. This contradicts the hardness of PAC learning.

7.3.1 Formal Proofs

In this section, we consider the non-uniform model (circuits) as a computational model for learners. First, we formally define RRHS-refutation in the non-uniform model as follows.

Definition 7.3.6 (RRHS-refutation: non-uniform)**.** *Let $\mathscr{C}$ be a concept class. We say a family of polynomial-size circuits $\{R_n\}_{n\in\mathbb{N}}$, called a refuter, RRHS-refutes $\mathscr{C}$ with $m := m(n)$ samples if for any $n \in \mathbb{N}$, $f \in \mathscr{C}_n$, and $x_1, \ldots, x_m \in \{0,1\}^n$, R_n satisfies that*

1. *$R_n(x_1, \ldots, x_m, f(x_1), \ldots, f(x_m)) = 0$;*
2. *$\Pr_{b\sim\{0,1\}^m}[R_n(u_1, \ldots, u_m, b) = 1] \geq 1/2$.*

We say that $\mathscr{C}$ is non-uniformly RRHS-refutable if there exist a polynomial m and a polynomial-size refuter for $\mathscr{C}$ with m samples.

Then the equivalence between learning and refutation also holds in the non-uniform model. It is easily shown by combining the original proof for the uniform model with Adleman's trick [Adl78] to remove randomness from a refuter.

Theorem 7.3.7 (PAC learning $\Leftrightarrow$ RRHS-refutation: non-uniform)**.** *A concept class $\mathscr{C}$ is non-uniformly PAC learnable if and only if $\mathscr{C}$ is non-uniformly RRHS-refutable.*

Proof of Theorem 7.3.2

Proof of Theorem 7.3.2. This theorem follows from Theorem 7.3.7 and a simple padding argument. As in the statement, let $\mathscr{C}$ be a concept class evaluated by a circuit family $\{C_n\}_{n\in\mathbb{N}}$, where $C_n : \{0,1\}^{s(n)} \times \{0,1\}^n \to \{0,1\}$ is a circuit of size at most $\tau(n)$.

For one direction, we assume that there exists c_0 such that $C_n \in \mathsf{Learnable}_{\tau,n^c}$ for any $c \geq c_0$ and n. Then for any $n \in \mathbb{N}$, there exists a circuit $L_n \in \mathsf{SIZE}[n^2]$ satisfying completeness and soundness in Definition 7.3.1 with $m = n^{c_0}$ samples. Notice that the family $\{L_n\}_{n\in\mathbb{N}}$ is a RRHS-refuter for $\mathscr{C}$ with m examples. Therefore, by Theorem 7.3.7, $\mathscr{C}$ is non-uniformly PAC learnable.

For the opposite direction, we assume that $\mathscr{C}$ is non-uniformly PAC learnable. By Theorem 7.3.7, $\mathscr{C}$ is also non-uniformly RRHS-refutable. Let $\{R_n\}_{n\in\mathbb{N}}$ be the polynomial-size refuter for $\mathscr{C}$ with m samples. Note that, if each R_n is contained in $\mathsf{SIZE}[n^2]$, then $C_n \in \mathsf{Learnable}_{\tau,m}$. In general, however, we can only assure that each R_n is contained in $\mathsf{SIZE}[n^a]$ for some $a \in \mathbb{N}$. We resolve this problem by the simple padding argument.

We only consider the case of $a > 2$, otherwise, $R_n \in \mathsf{SIZE}[n^2]$. For the size of example n and the sample size m, the length of input to the refuter is $nm + m = (n+1)m$. Therefore, the size of R_n is at most $N := ((n+1)m)^a$. Since m and N are polynomials in n, we can select $c_0 \in \mathbb{N}$ satisfying that for any $c \geq c_0$, $m \leq n^c$ and $N \leq ((n+1)n^c)^2 - (n+1)(n^c - m)$.

For any $c \geq c_0$, let R'_n be a refuter which takes n^c samples, executes R_n by using only m samples, and discards remaining $n^c - m$ samples. Since $\{R_n\}_{n\in\mathbb{N}}$ RRHS-refutes $\mathscr{C}$, $\{R'_n\}_{n\in\mathbb{N}}$ also RRHS-refutes $\mathscr{C}$. In addition, R'_n can be constructed from R_n and $(n+1)(n^c - m)$ additional input-gates, thus the size of R' is bounded above by $N + (n+1)(n^c - m) \leq ((n+1)n^c)^2$ and $R'_n \in \mathsf{SIZE}[n^2]$. This implies that $C_n \in \mathsf{Learnable}_{\tau,n^c}$ for any $c \geq c_0$. □

Proof of Theorem 7.3.3

Proof of Theorem 7.3.3. Fix polynomials τ and m arbitrary. The theorem follows from the definition of $\mathsf{Learnable}_{\tau,m}$ and the result by Lautemann [Lau83].

Let C be a circuit of size N computing an evaluation $E\colon \{0,1\}^{s(n)} \times \{0,1\}^n \to \{0,1\}$. If $N > \tau(n)$, then $C \notin \mathsf{Learnable}_{\tau,m}$. This trivial case is easily detected in polynomial-time. Therefore, we can assume that $N \leq \tau(n)$.

If $C \in \mathsf{Learnable}_{\tau,m}$, then there must exist a learner $L \in \mathsf{SIZE}[n^2]$ with m samples. We consider the description $\langle L \rangle$ as a witness and verify the following conditions in Π^p_2: for any examples $x_1, \ldots, x_m \in \{0,1\}^n$,

(1) for any $u \in \{0,1\}^{s(n)}$, $L(x_1, \ldots, x_m, E(u,x_1), \ldots, E(u,x_m)) = 0$;

(2) $\Pr_{b_1,\ldots,b_m}[L(x_1, \ldots, x_m, b_1, \ldots, b_m) = 1] \geq 1/2$.

The condition (1) is obviously checked in Π^p_1. For the condition (2), if we fix a learner L and examples $x_1, \ldots, x_m$, then it is easily checked in BPP by estimating the accepting probability of L. Since $\mathsf{BPP} \subseteq \Sigma^p_2 \cap \Pi^p_2$ [Lau83], we can replace the probabilistic part with Π^p_2. This concludes that $\mathsf{Learnable}_{\tau,m} \in \Sigma^p_3$.

In the above argument, if we can replace the probabilistic part with a deterministic polynomial-time verifier instead of Π^p_2, then we can also verify the condition (2) in Π^p_1. Therefore, $\mathsf{P} = \mathsf{BPP}$ implies $\mathsf{Learnable}_{\tau,m} \in \Sigma^p_2$. □

Proof of Theorem 7.3.5

In fact, we can show stronger statements than Theorem 7.3.5 as follows:

Lemma 7.3.8. *Let $\tau(n)$ be a polynomial satisfying $\tau(n) > n$. If $\mathsf{SIZE}[n^2]$ is non-uniformly PAC learnable, then $\mathsf{Learnable}_{\tau,n^c} \in \mathsf{P}$ for sufficiently large $c \in \mathbb{N}$.*

Proof. Suppose that $\mathsf{SIZE}[n^2]$ is non-uniformly PAC learnable. By Theorem 7.1.3 (for the non-uniform model) and Theorem 7.3.7, $\mathsf{SIZE}[\tau(n)]$ is non-uniformly RRHS-refutable. Let R be the refuter with n^{c_0} samples for some $c_0 \in \mathbb{N}$. By the same padding argument as the proof of Theorem 7.3.2, we can also assume that $R \in \mathsf{SIZE}[n^2]$.

For a given circuit $C\colon \{0,1\}^{s(n)} \times \{0,1\}^n \to \{0,1\}$, if the size of C is at most $\tau(n)$, then any target function is computed by a circuit of the size $\tau(n)$. Therefore, R satisfies the third condition in Definition 7.3.1. Thus, $C \in \mathsf{Learnable}_{\tau,n^c}$ holds for any $c \geq c_0$.

On the other hand, if the size of C is larger than $\tau(n)$, then C must not satisfy the second condition for $\mathsf{Learnable}_{\tau(n),n^c}$ for any c. In other words, for any $c \geq c_0$, $C \in \mathsf{Learnable}_{\tau(n),n^c}$ iff the size of C is at most $\tau(n)$. The latter condition is easily checked in polynomial-time from the description of C. Therefore, $\mathsf{Learnable}_{\tau,n^c} \in \mathsf{P}$ for any $c \geq c_0$. □

Lemma 7.3.9. *Let $\tau(n)$ be a polynomial satisfying $\tau(n) \geq n^{1+\epsilon}$ for some $\epsilon > 0$. If $\mathsf{SIZE}[n^2]$ is not non-uniformly PAC learnable, then $\mathsf{Learnable}_{\tau,n^c} \notin \mathsf{P}$ for any c.*

Proof. Let $s(n) = n + \sqrt{n}$ and $t(n) = (\text{the size of } UC_n^{s(n)}) = \tilde{O}(n + \sqrt{n} + n) = \tilde{O}(n)$. We assume that n is sufficiently large to satisfy that $t(n) + n + 1 \leq n^{1+\epsilon} \leq \tau(n)$.

Suppose that $\mathsf{SIZE}[n^2]$ is not non-uniformly PAC learnable. By Theorem 7.1.3 (for the non-uniform model), $\mathsf{SIZE}[s(n)]$ is also not non-uniformly PAC learnable. By Definition 7.3.1, if $u \in \mathsf{Learnable}_{\tau,n^c}$, then $u \in \mathsf{Learnable}_{\tau,n^{c+1}}$ for any $c \in \mathbb{N}$ and $u \in \{0,1\}^*$. Therefore, by Theorem 7.3.2, we have that

$$UC_n^{s(n)} \notin \mathsf{Learnable}_{\tau,n^c} \text{ for any } c \in \mathbb{N}.$$

Suppose for contradiction that there exists $c \in \mathbb{N}$ such that $\mathsf{Learnable}_{\tau,n^c} \in \mathsf{P}$ and M be the polynomial-time algorithm for $\mathsf{Learnable}_{\tau,n^c}$.

Now we construct an algorithm A solving the circuit-SAT problem as follows: on input $\langle C \rangle \in \{0,1\}^n$ (assume that the size of C is at most n and the input length is $\ell \leq n$), (1) A constructs another circuit $C'\colon \{0,1\}^{\ell+e(s(n))} \times \{0,1\}^n \to \{0,1\}$ as $C'(u \circ u', x) = C(u) \wedge UC_n^{s(n)}(u', x)$ where $u \in \{0,1\}^\ell$, $u' \in \{0,1\}^{e(s(n))}$, and $x \in \{0,1\}^n$, and (2) A outputs the value of $\neg M(C')$.

Note that the size of C' is at most $n + t(n) + 1 \leq \tau(n)$. If the given C is satisfiable, then there exists an assignment $u^* \in \{0,1\}^\ell$ such that $C(u^*) = 1$. By applying the partial assignment u^*, C' boils down to $UC_n^{s(n)}$. Remember that $UC_n^{s(n)} \notin \mathsf{Learnable}_{\tau,n^c}$. Therefore, $C' \notin \mathsf{Learnable}_{\tau,n^c}$, and $A(C)$ outputs $\neg M(C') = 1$ for any satisfiable circuit C.

On the other hand, if C is not satisfiable, then $C'(u \circ u', x) \equiv (0 \wedge UC_n^{n^2}(u', x)) \equiv 0$. Since we can easily RRHS-refute the class $\{0\}$ (with 2 samples by calculating negation of OR of two labels), $C' \in \mathsf{Learnable}_{\tau,n^c}$. Therefore, $A(C)$ outputs $\neg M(C') = 0$ for any unsatisfiable circuit C.

Therefore, A correctly solves the circuit-SAT problem, and $\mathsf{P} = \mathsf{NP}$. This implies $\mathsf{SIZE}[n^2]$ is (even uniformly) PAC learnable by Corollary 2.6.3. This contradicts the assumption. □

Lemmas 7.3.8 and 7.3.9 immediately derive Theorem 7.3.5 (by setting $\tau(n) = n^2$) and the following theorem. Theorem 7.3.10 shows that the specific setting of $\tau(n)$ in Definition 7.3.4 does not affect the asymptotic complexity of our meta-PAC learning problem.

Theorem 7.3.10. *Let $\tau(n)$ be a polynomial satisfying $\tau(n) \geq n^{1+\epsilon}$ for some $\epsilon > 0$. Meta-PAC learning is hard iff* $\mathsf{Learnable}_{\tau,n^c} \notin \mathsf{P}$ *for infinitely many $c \in \mathbb{N}$.*

Chapter 8

On Basing Auxiliary-Input Cryptography on NP-Hardness

In this chapter, we consider whether *auxiliary-input* cryptographic primitives can be based on NP-hardness in a common framework of proofs. In the previous chapters, we discussed several types of auxiliary-input cryptographic primitives, which are natural intermediate notions between the worst-case hardness and the standard cryptographic primitives and also have deep connections to the hardness of PAC learning. Therefore, basing auxiliary-input cryptographic primitives on NP-hardness is a natural intermediate step towards basing the standard cryptographic primitives or the hardness of learning on NP-hardness. Particularly, in Section 7.2, we introduced an auxiliary-input hitting set generator (AIHSG) and proved that the existence is *implied* by the hardness of PAC learning. Thus, basing AIHSG on NP-hardness is valuable as a good test case for NP-hardness of PAC learning, which is a longstanding open question in computational learning theory.

8.1 Background

A central tool for basing cryptographic primitives on hardness of a language L is a *reduction*[1], i.e., the way to translate recognizing L into breaking a cryptographic primitive. A reduction is a powerful proof technique even if it is restricted to a simple form, and in fact, a nonadaptive black-box (BB) reduction is sufficient to show many brilliant results and has played a crucial role in theoretical computer science. Furthermore, in general, a nonadaptive BB reduction is more accessible than adaptive or non-black-box ones. Therefore, it is a natural attempt to apply such a familiar proof technique even for constructing secure cryptographic primitives.

Unfortunately, Bogdanov and Trevisan [BT06b] gave strong evidence that such a simple reduction is insufficient for cryptography based on NP-hardness. In general, breaking security of cryptographic primitives is formulated as an NP problem on an efficiently samplable distribution fixed in advance. They showed that there is no nonadaptive BB reduction from an NP-hard problem to such a distributional NP problem unless the polynomial hierarchy collapses. Therefore, as a corollary, their work excluded the attempt to apply nonadaptive BB reductions for basing cryptography under the reasonable assumption on the polynomial hierarchy. Moreover, subsequent work gave

[1] It is often called a *security* reduction to distinguish it from another type of reduction from computing some primitive P to another primitive Q (see [RTV04]).

stronger consequences in more specific cases of basing several cryptographic primitives [AGGM06; GV08; ABX08; HMX10; BL13; BB15; LV16; HW20].

Along this line of research, we investigate whether nonadaptive BB reduction is sufficient to base auxiliary-input cryptographic primitive on NP-hardness. The importance of this work is to extend our current knowledge on the central proof technique of nonadaptive BB reductions out of the previous worst-case-to-average-case framework and to identify the inherent difficulty on constructing cryptographic primitives on NP-hardness more finely. Remember that an auxiliary-input cryptographic primitive is defined as a family of primitives indexed by the auxiliary-input and has a relaxed security requirement: at least one primitive in the family is required to be secure depending on each adversary (instead of one fixed primitive secure against all adversaries). Namely, adversaries must break all primitives in the worst-case sense on auxiliary-input, and the task is not directly formulated as a distributional NP-problem because the distribution is not uniquely determined beforehand due to the auxiliary-input. Thus the previous work on distributional NP problem cannot be directly applied to auxiliary-input cryptography.

Now let us mention previous studies related to nonadaptive BB reductions to auxiliary-input cryptography. Applebaum, Barak, and Xiao [ABX08] observed that we cannot apply a nonadaptive fixed-auxiliary-input BB reduction, which is a restricted nonadaptive BB reduction given access to only one auxiliary-input, unless the polynomial hierarchy collapses. However, the restricted access to auxiliary-input seems to be too strict and it implicitly yields a reduction from an NP-hard language to some fixed cryptographic primitive (depending on the instance). In fact, the above result was shown in almost the same way to the previous result by Akavia, Goldreich, Goldwasser, and Moshkovitz [AGGM06] for standard cryptographic primitives. The same work and later Xiao [Xia09a] observed that generalizing their result to nonadaptive BB reductions seems hard by giving the explicit technical issue. To the best of our knowledge, we have no negative result on general nonadaptive BB reductions to base auxiliary-input cryptography on NP-hardness before this work.

8.2 Our Results

Our main theorem is informally stated as follows.

Theorem (informal)**.** *If there is a nonadaptive BB reduction from an* NP*-hard language L to breaking an auxiliary-input cryptographic primitive P, then according to the type of P we have that:*

- *If P is an auxiliary-input pseudorandom generator, then the polynomial hierarchy collapses;*
- *If P is an auxiliary-input one-way function or an auxiliary-input hitting set generator, then there is also an adaptive reduction from L to inverting an infinitely-often one-way function.*

The first result shows strong evidence that auxiliary-input pseudorandom generators (AIPRG) cannot be based on NP-hardness via nonadaptive BB reductions as standard cryptography. The second result shows that a nonadaptive BB reduction for basing the other auxiliary-input primitives on NP-hardness yields another surprising consequence that one-way function whose security is based on NP-hardness. Again, we remark that an auxiliary-input hitting set generator (AIHSG) is much weaker primitive than standard cryptographic primitives whose existence is even weaker than the hardness of PAC learning. What is surprising is that even a nonadaptive BB reduction for such a weak primitive is indeed sufficient for excluding both Heuristica and Pessiland! Note that this does

not contradict the previous barrier result against nonadaptive BB reductions for basing OWF on NP-hardness [AGGM06] because the resulting reduction in our theorem is *adaptive*.

The second result is not sufficient to exclude nonadaptive BB reductions for basing auxiliary-input primitives on NP-hardness, and it has two opposite interpretations. However, let us stress that both interpretations are quite nontrivial and yield new knowledge about nonadaptive BB reductions. One interpretation is a pessimistic (or realistic) one. As mentioned in the introduction, no one has not come up with the construction of a one-way function based on NP-hardness for several decades despite its importance. Thus, this result is still strong evidence of difficulty finding such a simple reduction. The other interpretation is an optimistic one as a new approach to constructing a one-way function. We will further discuss this optimistic perspective and its novelty in Section 8.2.2.

A reader who is familiar with cryptography may wonder why the consequences are different between an auxiliary-input one-way function (AIOWF) and AIPRG. In fact, AIPRG is constructed from any AIOWF by applying the known BB construction of a pseudorandom generator from a one-way function. However, if such construction requires an adaptive security proof, then the property on nonadaptive is lost in translating reductions for AIOWF into reductions for AIPRG via the adaptive security reduction. To the best of our knowledge, all currently known constructions of pseudorandom generators [HILL99; Hol06; VZ12; HRV13] use adaptive techniques in the security proof; e,g, construction of false entropy generators [HILL99], the uniform hardcore lemma [Hol05], and the uniform min-max theorem [VZ13]. This technical issue prevents us from applying the first result for AIPRG to AIOWF. For a similar reason, our second result on AIOWF is incomparable with the previous work on basing hardness of learning by Applebaum, Barak, and Xiao [ABX08][2].

8.2.1 Formal Descriptions

Now, we formally state the main results. For each auxiliary-input cryptographic primitives (i.e., AIPRG, AIOWF, and AIHSG), we first define the BB security reduction and then present the formal statement of the theorem. Note that, in this chapter, we implicitly assume that an auxiliary-input function is polynomial-time computable.

A BB reduction for AIPRG is defined as follows. It is easily verified that the following BB reduction from a language L to distinguishing an auxiliary-input function G shows that G is an AIPRG if $L \notin \mathsf{BPP}$.

Definition 8.2.1 (Black-box reduction to distinguishing auxiliary-input function)**.** *Let L be a language and $G := \{G_z : \{0,1\}^n \to \{0,1\}^{\ell(n)}\}_{z\in\{0,1\}^*}$ be an auxiliary-input function with $\ell(n) > n$. We say that there exists a black-box (BB) reduction from L to distinguishing G if for any polynomial p, there exists a randomized polynomial-time oracle machine $R^?$ such that for any oracle $\mathcal{O}$ that $(1/p)$-distinguishes G, i.e., for every auxiliary-input $z \in \{0,1\}^*$*

$$\left| \Pr_{\mathcal{O},U_n}[\mathcal{O}(z, G_z(U_n)) = 1] - \Pr_{\mathcal{O},U_{\ell(n)}}[\mathcal{O}(z, U_{\ell(n)}) = 1] \right| \geq 1/p(n),$$

and for any $x \in \{0,1\}^*$, R satisfies that

$$\Pr_R[R^{\mathcal{O}}(x) = L(x)] \geq 2/3.$$

[2] Although the hardness of learning is conceptually weaker than AIOWF, their work used the property of black-box in the formulation of a reduction to learning and indeed yielded a reduction to inverting AIOWF in the end.

Moreover, we say that there exists a nonadaptive BB reduction from L to distinguishing G if all R make its queries independently of any answer by oracle for previous queries.

Now we present the first main result on AIPRG.

Theorem 8.2.2. *For any auxiliary-input function $G = \{G_z : \{0,1\}^n \to \{0,1\}^{\ell(n)}\}_{z\in\{0,1\}^*}$ with $\ell(n) > n$, there exists no nonadaptive BB reduction from an NP-hard language L to distinguishing G unless the polynomial hierarchy collapses.*

A BB reduction for AIOWF is defined as follows. It is easily verified that for any polynomial p, the following BB reduction from a language L to $(1-1/p)$-inverting an auxiliary-input function f shows that f is an AIOWF if $L \notin \mathsf{BPP}$.

Definition 8.2.3 (Black-box reduction to inverting auxiliary-input function)**.** *Let L be a language, p be a polynomial, and $f := \{f_z : \{0,1\}^n \to \{0,1\}^\ell\}_{z\in\{0,1\}^*}$ be an auxiliary-input function. We say that a randomized polynomial-time oracle machine $R^?$ is a black-box (BB) reduction from L to $(1-1/p)$-inverting f if for any oracle $\mathcal{O}$ that $(1-1/p)$-inverts f, i.e., for every $z \in \{0,1\}^*$,*

$$\Pr_{\mathcal{O},U_n}[\mathcal{O}(z, f_z(U_n)) \in f_z^{-1}(f_z(U_n))] \geq 1 - 1/p(n),$$

and for any $x \in \{0,1\}^*$, R satisfies that

$$\Pr_R[R^{\mathcal{O}}(x) = L(x)] \geq 2/3.$$

Moreover, we say that R is nonadaptive if all R's queries are made independently of any answer by oracle for previous queries.

Now we present the second main result on AIOWF.

Theorem 8.2.4. *For any auxiliary-input function $f = \{f_z : \{0,1\}^n \to \{0,1\}^\ell\}_{z\in\{0,1\}^*}$ and polynomial p, if there exists a nonadaptive BB reduction from an NP-hard language L to $(1-1/p)$-inverting f, then $\mathsf{NP} \nsubseteq \mathsf{BPP}$ also implies that a one-way function exists (via an adaptive BB reduction).*

We review the definition of AIHSG because we consider the general largeness function γ in this chapter (it was fixed to $1/2$ in Chapter 7).

Definition 8.2.5 (Auxiliary-input hitting set generator)**.** *Let $G = \{G_z : \{0,1\}^n \to \{0,1\}^{\ell(n)}\}_{z\in\{0,1\}^*}$ be an auxiliary-input function. For a function $\gamma : \mathbb{N} \to (0,1)$, we say that a randomized adversary A γ-avoids G if for any (public) auxiliary-input $z \in \{0,1\}^*$ and (private) input $x \in \{0,1\}^{n(|z|)}$,*

$$\Pr_A[A(z, G_z(x)) = 0] \geq 2/3 \ \text{ and } \Pr_{y\sim\{0,1\}^{\ell(n(|z|))}}\left[\Pr_A[A(z,y) = 1] \geq 2/3\right] \geq \min(\gamma(n), \tau_z),$$

where τ_z be a trivial limitation[3] defined as $\tau_z = 1 - \frac{|G_z(\{0,1\}^n)|}{2^{\ell(n)}}$.

We say that G is a γ-secure auxiliary-input hitting set generator (AIHSG) if $\ell(n) > n$ and there exists no polynomial-time randomized algorithm $(1-\gamma)$-avoiding G.

[3]In this paper, we consider general settings of γ and ℓ. Thus, we adopted the trivial limitation in the definition to avoid arguing about invalid settings where γ-avoiding the generator is impossible by definition.

Although it is easily verified that AIPRG is also AIHSG (with any security $\gamma(n) = 1/\mathsf{poly}(n)$), the opposite direction is open at present. In fact, the hardness of learning implies the existence of AIHSG; by contrast, we must overcome the barrier by oracle separation to show the existence of AIPRG (equivalently, AIOWF) from the hardness of learning [Xia09b]. Thus, AIHSG seems to be a much weaker notion than AIOWF and AIPRG under our current knowledge.

A BB reduction for AIHSG is defined as follows. It is easily verified that the following BB reduction from a language L to $(1-\gamma)$-avoiding an auxiliary-input function G shows that G is a γ-secure AIHSG if $L \notin \mathsf{BPP}$.

Definition 8.2.6 (Black-box reduction to avoiding auxiliary-input function)**.** *Let L be a language, γ be a function, and $G := \{G_z : \{0,1\}^n \to \{0,1\}^\ell\}_{z\in\{0,1\}^*}$ be an auxiliary-input function. We say that a randomized polynomial-time oracle machine $R^?$ is a black-box (BB) reduction from L to $(1-\gamma)$-avoiding G if for any oracle $\mathcal{O}$ that $(1-\gamma)$-avoids G and $x \in \{0,1\}^*$, R satisfies that*

$$\Pr_R[R^{\mathcal{O}}(x) = L(x)] \geq 2/3.$$

Moreover, we say that R is nonadaptive if all R's queries are made independently of any answer by oracle for previous queries.

Now we present the third main result on AIHSG.

Theorem 8.2.7. *Let p be a polynomial and $G := \{G_z : \{0,1\}^n \to \{0,1\}^{\ell(n)}\}_{z\in\{0,1\}^*}$ be an auxiliary-input function where $\ell(n) > (1+\epsilon)\cdot n$ for some constant $\epsilon > 0$. If there exists a nonadaptive BB reduction from an NP-hard language L to $(1-1/p)$-avoiding G, then $\mathsf{NP} \nsubseteq \mathsf{BPP}$ also implies that a one-way function exists (via an adaptive BB reduction).*

8.2.2 Discussion and Future Directions

As mentioned in Section 8.1, Theorems 8.2.4 and 8.2.7 are also regarded as approaches to construct one-way functions whose security is based on NP-hardness. In this section, we discuss the novelty of this optimistic perspective and propose future directions, including the investigation of the validity.

Our results are rephrased as follows: If we could connect NP-hardness to some auxiliary-input primitives (i.e., AIOWF or AIHSG) with a novel (nonadaptive BB) reduction, then we can automatically extend the connection to standard cryptographic primitives, particularly OWF. At present, the latter task of removing auxiliary-input from primitives seems to be quite non-trivial. Particularly, in Section 9.2, we give a strong oracle separation between AIOWF and OWF.

Theorem 8.2.8. *There exists an oracle relative to which there is no one-way function but there exists an auxiliary-input one-way function secure against nonuniform $2^{O(n/\log n)}$-time adversaries.*

Thus, we cannot hope any relativizing proof to remove auxiliary-input from primitives. Additionally, there are several barriers by other oracle separations at the intermediate levels to base OWF on NP-hardness [e.g., Xia09b; Imp11]. Although such barriers on relativization are common throughout theoretical computer science [e.g. BGS75], there are only a few success stories of overcoming such barriers at present. Unfortunately, Theorems 8.2.4 and 8.2.7 do not give any solution to break these barriers, and a new non-relativized technique is still required. Specifically, if a nonadaptive BB reduction to AIOWF or AIHSG is also relativized[4], then our proof also yields relativized reductions that contradict Theorem 8.2.8 or other oracle separations in Chapter 9.

[4]Note that oracle separations do not necessarily rule out BB reductions form particular languages, not as fully BB reductions defined by Reingold, Trevisan, and Vadhan [RTV04].

Nevertheless, we claim that our result gives one hopeful insight. Although there seems to be several barriers towards cryptography based on NP-hardness, the essential barrier we must overcome might be few. Theorems 8.2.4 to 8.2.8 certainly show that if we could find a non-relativized breakthrough at an intermediate level toward cryptography (i.e., auxiliary-input primitives), then it will solve other challenging open questions at the higher level, including a worst-case to average-case reduction for NP, an errorless to error-prone reduction for DistNP, and a construction of OWF whose security is based on the average-case hardness of NP. From this perspective, we conjecture that the hardness of basing OWF on NP-hardness might heavily rely on a much smaller part at an intermediate level. This conjecture seems to be somewhat controversial but enhances the significance of further investigation on basing auxiliary-input or other intermediate cryptographic primitives instead of standard ones.

The discussion above leads to the following two possible directions. The first direction is to find other scenarios where a breakthrough at an intermediate level also brings benefits at the higher level. This direction might reduce constructing standard cryptographic primitives to the task at the low level and give new insights into complexity-based cryptography. The second direction is to refute such an attempt on intermediate primitives with convincing evidence if it gives the wrong direction. Particularly, in our case, there is a possibility that nonadaptive BB reductions to base AIOWF and AIHSG on NP-hardness indeed yield the collapse of the polynomial-hierarchy as in the case of AIPRG. For the second direction, we list two concrete ways: (1) finding a new construction of AIPRG from AIOWF with nonadaptive security proof; and (2) generalizing the previous results for OWF [AGGM06] or HSG [HW20] to each auxiliary-input analog for the stronger consequence. At least the latter approach seems to require some new technique to simulate nonadaptive BB reductions, as observed by Applebaum, Barak, and Xiao [ABX08] and Xiao [Xia09a].

8.3 Overview of Proof Ideas

In this section, we present proof ideas of Theorems 8.2.2 to 8.2.7. Note that Theorem 8.2.7 heavily relies on Theorem 8.2.4, and Theorem 8.2.4 heavily relies on Theorem 8.2.2. Therefore, although each proof idea may look pretty simple and intuitive, our construction of OWF for Theorem 8.2.7 becomes complicated and quite non-trivial as a whole.

8.3.1 On Basing AIPRG on NP Hardness: Proof Idea of Theorem 8.2.2

The crucial part of the proof is to give a construction of HSG from AIPRG with a nonadaptive BB security reduction from distinguishing AIPRG to avoiding HSG. Note that such an implication has been implicitly shown in [HS17]. For completeness, we will give an explicit and simpler construction to show the same implication. Although the reader may think that our construction is too fundamental and looks somewhat trivial, to the best of our knowledge, no one has mentioned such a clear relationship between AIPRG and HSG.

First, we explain why a nonadaptive BB security reduction from distinguishing AIPRG to avoiding HSG is crucial. We can easily observe that avoiding HSG is directly formulated as the following distributional NP problem in the errorless setting: for uniformly random instance y, determine whether y is contained in the image of HSG. Therefore, the nonadaptive BB reduction from distinguishing AIPRG to avoiding HSG also yields a nonadaptive BB reduction from distinguishing AIPRG to the distributional NP problem. Thus, any nonadaptive BB reduction from an NP-hard problem to distinguishing AIPRG in the assumption indeed yields a nonadaptive BB reduction from

the same NP-hard problem to the distributional NP problem. By the previous result by Bogdanov and Trevisan [BT06b], such a reduction implies the collapse of the polynomial-hierarchy.

Our construction of HSG from AIPRG is the following: just considering the both of auxiliary-input and input to AIPRG as usual input to HSG. More specifically, let $G = \{G_z : \{0,1\}^n \to \{0,1\}^{\ell(n)}\}_{z\in\{0,1\}^*}$ be an AIPRG. Then the construction of HSG G' is given as $G'(z \circ x) = G_z(x)$. Note that, when $z + n(|z|) > \ell(n(|z|))$ holds, G' does not satisfy the syntax on stretching input. In the formal proof, therefore, we first stretch the output of G by the standard technique in cryptography. We can easily check that the security reduction for this stretching (shown by the famous hybrid argument) is nonadaptive.

Let $\gamma(n)$ be a reciprocal of polynomial. The security reduction from γ-avoiding G' to distinguishing G is also simple: just considering an adversary A for G' as an adversary for G. Obviously, this reduction is nonadaptive. To see the correctness, assume that A γ-avoids G'. For simplicity, we also assume that A is deterministic and $\gamma(n) < \tau_n$. Whenever the input y is pseudorandom string contained in the image of G', $A(y)$ does not output 1. On the other hand, when y is a truly random string, then $A(y)$ outputs 1 with probability at least $\gamma(n)$. Thus, A can distinguish the uniform distribution from all distributions on the image of G' with an advantage at least $\gamma(n)$. For any auxiliary-input z, $G_z(U_{n(|z|)})$ is distributed on the image of G'. Thus, A also γ-distinguishes G.

8.3.2 On Basing AIOWF on NP Hardness: Proof Idea of Theorem 8.2.4

To focus on the idea, we omit all arguments about the success probabilities of adversaries in this section. First, let us prepare several reductions. Let $R_{L\to f}$ be the nonadaptive BB reduction from L to inverting f in the assumption. By the construction of PRG from OWF [e.g., HILL99], there exist an auxiliary-input generator G and an adaptive BB reduction $R_{f\to G}$ from inverting f to distinguishing G. By the result in Section 8.3.1, there exist an NP-language L' and a nonadaptive BB reduction $R_{G\to L'}$ from distinguishing G to a distributional NP problem (L', U) in the errorless setting. Since $L' \in \mathsf{NP}$ and L is NP-hard, there exists a Karp reduction $R_{L'\to L}$ from L' to L.

Now we consider the following procedure:

1. select an instance x' of L' at random;
2. translate x' into an instance x of L as $x = R_{L'\to L}(x')$;
3. plug x into $R_{L\to f}$ with a random tape r;

At this stage, $R_{L\to f}$ makes polynomially many queries $(z_1, y_1), \ldots, (z_q, y_q)$.

4. answer the queries by some inverting oracle $\mathcal{O}$;
5. output the same decision $b \in \{0,1\}$ as $R_{L\to f}$.

Note that if the oracle $\mathcal{O}$ correctly inverts f, then the resulting decision b is $L(x)$ with high probability, and $L(x)$ is equal to $L'(x')$.

The crucial observation is that there is no worst-case sense at all in the above procedure because both x' and r are selected at random. Therefore, all queries at the stage 3 are indeed efficiently samplable, and the inverting oracle no longer needs to invert f for every auxiliary-input at the stage 4. This observation leads to our construction of a standard OWF g.

The function g takes three inputs x', r, and x^f, which intuitively represents a random instance of L', randomness for $R_{L\to f}$, and input for f, respectively. Then $g(x', r, x^f)$ imitates the above procedure as follows: (2') translate x' into an instance x of L as $x = R_{L'\to L}(x')$, (3') plug x into $R_{L\to f}$ with randomness r, then randomly pick one of auxiliary-input z in queries by $R_{L\to f}$ and output $f_z(x^f)$.

We will show that the above-mentioned g is one-way if $\mathsf{NP} \not\subseteq \mathsf{BPP}$. Suppose for contradiction that there exists an adversary A inverting g. Remember that g simulates a distribution on auxiliary-input in the procedure above. Thus, intuitively, we can replace the inverting oracle $\mathcal{O}$ with the adversary A at the stage 4 with high probability in the execution of $R_{L\to f}$. In fact, this is a little technical part, and we will give further detail in Section 8.5. Then the procedure above no longer needs any oracle and yields a randomized algorithm solving (L', U) on average. By applying reductions $R_{G\to L'}$, $R_{f\to G}$, and $R_{L\to f}$ in this order, this also yields a randomized polynomial-time algorithm for L. Since L is NP-hard, we conclude that $\mathsf{NP} \subseteq \mathsf{BPP}$.

We remark that $R_{G\to L'}$ is a nonadaptive BB reduction thanks to our simple construction in Section 8.3.1. Therefore, if we also have a construction of AIPRG G from AIOWF f with a nonadaptive BB reduction from inverting f to distinguishing G, then the proof above leads to a nonadaptive BB reduction from L to (L', U), which implies the collapse of the polynomial hierarchy as in Theorem 8.2.2. Thus, finding a construction of AIPRG with such a simple security reduction is one direction for excluding a nonadaptive BB reduction to base AIOWF on NP-hardness, as mentioned in Section 8.2.2.

8.3.3 On Basing AIHSG on NP Hardness: Proof Idea of Theorem 8.2.7

The key idea for the proof is to classify each query generated by the nonadaptive BB reduction (in the theorem) into a "light" query and a "heavy" query. A similar technique was also used in the previous work for HSG by Gutfreund and Vadhan [GV08] and Hirahara and Watanabe [HW20]. We first see the previous case of HSG and then explain the difference to our case of AIHSG.

The Case of Hitting Set Generator (Previous work)

Let $G : \{0,1\}^n \to \{0,1\}^{\ell(n)}$ be a generator with $\ell(n) \geq (1 + \Omega(1)) \cdot n$ and $R^?$ be a nonadaptive BB reduction from an NP-language L to avoiding G. Without loss of generality, we can assume that all marginal distributions on queries by R are identical regardless of each query position by applying a random permutation on queries before asking them to oracle. Thus, for each input $x \in \{0,1\}^n$, one distribution Q_x on queries is determined. We choose a threshold (roughly) $\tau = 1/\tilde{\Theta}(2^n)$ and define a light (resp. heavy) query $y \in \{0,1\}^{\ell(n)}$ as a query generated according to Q_x with probability less (resp. greater) than the threshold τ.

The essential part of the proof is to simulate the avoiding oracle for G by using the classification of queries. First, assume that we could (somehow) distinguish the heavy case and the light case for a given query. Then we can simulate one of avoiding oracles simply as follows: for each query y generated by $R(x)$, (1) determine whether y is heavy or light; (2) answer 0 (resp. 1) if y is heavy (resp. light) query. Let $\mathcal{O}'$ be the induced oracle by the strategy above. Note that the probability that $\mathcal{O}'(y)$ outputs 0 is exponentially small because the fraction of heavy query is $\tilde{\Theta}(2^n)/2^{\ell(n)} \leq 2^{-\Omega(1)n}$ for the length $\ell(n)$ of query. Thus, $\mathcal{O}'$ satisfies the condition on the probability of outputting 1. However, $\mathcal{O}'$ is not a valid avoiding oracle for G because possibly there is a query y such that y is light but contained in $\mathrm{Im}G$. In this case, $\mathcal{O}'(y)$ outputs 1 for $y \in \mathrm{Im}G$.

The key observation to overcome this issue is the following:

> ($\star$) For each length $\ell(n)$ of query (where the input size is n), the size of $\mathrm{Im}G$ is at most 2^n; thus the probability that R asks some light query contained in $\mathrm{Im}G$ (that is, "bad" query) is bounded above by $2^n/\tilde{\Theta}(2^n) \leq 1/\mathsf{poly}(n)$.

Therefore, $\mathcal{O}'$ is consistent with some avoiding oracle, and $R^{\mathcal{O}'}(x)$ correctly recognizes x with high probability over the execution of R.

By the above argument, we can reduce avoiding a generator to distinguishing heavy and light queries. For the latter task, Gutfreund and Vadhan [GV08] gave a $\mathsf{BPP}^{\mathsf{NP}}$ algorithm by approximation of counting by Jerrum, Valiant, and Vazirani [JVV86], and Hirahara and Watanabe [HW20] gave an $\mathsf{AM} \cap \mathsf{coAM}$ algorithm by the generalized version of the protocol by Feigenbaum and Fortnow [FF93].

The Case of Auxiliary-input Hitting Set Generator (Our work)

Now we move on to our case of AIHSG. Let $G = \{G_z : \{0,1\}^{n(|z|)} \to \{0,1\}^{\ell(n(|z|))}\}_{z \in \{0,1\}^*}$ be an auxiliary-input generator with $\ell(n) \geq (1 + \Omega(1)) \cdot n$ and $R^?$ be a nonadaptive BB reduction from an NP-language L to avoiding G. We can also assume that all query distributions of $R^?(x)$ are identical to Q_x regardless of query position.

On applying the above argument to our case of AIHSG, the problematic part is the key observation ($\star$). Remember that an adversary for AIHSG must avoid G_z for all $z \in \{0,1\}^*$, and auxiliary-input is possibly longer than output. Therefore, we cannot bound the size of the image of the generator in general because the image may span the whole range (for example, consider the following generator $G_z(x) = z \oplus (x \circ 0^{|z|-|x|})$ for $|z| > n(|z|)$).

To resolve this, we need to consider each case of auxiliary-input z separately. Therefore, we change the definitions of "light" and "heavy" and let them adapt to auxiliary-input. Let $p_x(z)$ be a probability that Q_x generates a query of auxiliary-input z. If we can bound the probability that R makes light query (z, y) with $y \in \mathrm{Im}G_z$ by $1/(\mathsf{poly}(n) \cdot p_x(z))$ for any z, then R makes such a "bad" query (z, y) with probability at most $\sum_z p_x(z) \cdot 1/(\mathsf{poly}(n) \cdot p_x(z)) = 1/\mathsf{poly}(n)$. Then we can use the same argument in the case of HSG and reduce avoiding G to distinguishing heavy and light cases. This idea naturally leads to the following new definition of "light" and "heavy": separating each query (z, y) by the conditional probability $p_x(y|z)$ that y is asked conditioned on the event that z is asked. In fact, as shown in Section 8.6, this modification will work well even for AIHSG.

However, one issue remains: how can we distinguish heavy and light queries? To this end, we must verify the largeness of the conditional probability of the given query. This part essentially prevents us from applying the previous results. Since we consider a polynomial-time-computable generator, the simulation with NP oracle does not give any nontrivial result, not as the work by [GV08][5]. Even for the simulation in $\mathsf{AM} \cap \mathsf{coAM}$ by Hirahara and Watanabe [HW20], there seem to be several technical issues. We cannot trivially verify the size of conditional probability by such protocols due to the restricted use of the upper bound protocol [AH91]. Moreover, we cannot possibly even sample the conditional distribution efficiently for fixed auxiliary-input (for example, consider the query distribution on $((z, vk), y)$ where y is a secure signature to z verified with a public-key vk).

[5]Their work concerned the original aim of HSG, that is, derandomization [e.g., IW01]. For this purpose, we consider (possibly) exponential-time-computable HSG G, and avoiding G in $\mathsf{BPP}^{\mathsf{NP}}$ is quite nontrivial. However, in our case where G is polynomial-time-computable, avoiding G is in NP trivially.

Our idea is to adopt the approximation method for probability by Impagliazzo and Levin [IL90], i.e., the algorithm in Section 4.6. Remember that the approximation method enables to reduce approximating the probability $p_y = \Pr_{U_n}[y = f(U_n)]$ to inverting f for any polynomial-time computable f under the situation where $y = f(x)$ and $x \in \{0,1\}^n$ is selected at random. In fact, the approximation method holds even for auxiliary-input function by the same proof of Theorem 4.6.1 (note that a similar idea was also used in [OW93]). By using the approximation algorithm for each circuit sampling query and auxiliary-input, we have a good approximation of $p_x(y|z)$ for query (z, y) generated by $R^?(x)$. Thus, the approximation algorithm enables us to classify the given (z, y) correctly. Note that the auxiliary-input in the approximation algorithm essentially corresponds to the input x for each circuit sampling query and auxiliary-input.

To show Theorem 8.2.7, we need further observations. Since R makes its queries nonadaptively, we can also invoke the approximation algorithm nonadaptively. Moreover, the approximation algorithm indeed uses an inverting adversary for a certain AIOWF as black-box and nonadaptively, as seen in the proof of Theorem 4.6.1. As a result, a nonadaptive BB reduction from an NP-hard language L to avoiding AIHSG yields a nonadaptive BB reduction from L to inverting AIOWF. Thus, by Theorem 8.2.4, R also yields a one-way function under the assumption that $\mathsf{NP} \nsubseteq \mathsf{BPP}$.

8.4 On Basing Auxiliary-Input Pseudorandom Generator on NP-Hardness

In this section, we formally rule out nonadaptive BB reductions from an NP-hard problem to distinguishing AIPRG based on Section 8.3.1. Let us state the main theorem again.

Theorem (Reminder of Theorem 8.2.2)**.** *For any auxiliary-input function $G = \{G_z : \{0,1\}^n \to \{0,1\}^{\ell(n)}\}_{z\in\{0,1\}^*}$ with $\ell(n) > n$, there exists no nonadaptive BB reduction from an NP-hard language L to distinguishing G unless the polynomial hierarchy collapses.*

Notations. For $n, k \in \mathbb{N}$, $x \in \{0,1\}^n$, and $n_1, \ldots, n_k \in [n]$ with $\sum_i n_i = n$, we use the notation $x \to_{n_1,\ldots,n_k} (x^{(1)}, \ldots, x^{(k)})$ to refer to the separation of x into k substrings satisfying that $x = x^{(1)} \circ \cdots \circ x^{(k)}$ and $|x^{(i)}| = n_i$ for each $i \in [k]$.

First, we give the nonadaptive BB reduction from distinguishing AIPRG to avoiding HSG.

Lemma 8.4.1. *Let G be an auxiliary-input function stretching its input, and let $m : \mathbb{N} \to \mathbb{N}$ be an arbitrary polynomial. There exists a polynomial-time computable function $G' : \{0,1\}^n \to \{0,1\}^{m(n)}$ and a randomized polynomial-time oracle machine $R^?$ satisfying the following: for any polynomial γ', there exists a polynomial γ such that for any oracle $\mathcal{O}$ which $1/\gamma'$-avoids G', $R^{\mathcal{O}}$ $1/\gamma$-distinguishes G. Moreover, $R^?$ is nonadaptive.*

Proof. Without loss of generality, we can assume that G has the stretch $\ell(n) = n+1$ by discarding the suffix of output. We define the generator $G' : \{0,1\}^{n'} \to \{0,1\}^{m(n')}$ in the lemma as

$$G'(x) = \begin{cases} b_1 \circ \cdots \circ b_{m(n)} & \text{if } (n' =)|x| = a + n(a) \text{ for some } a \in \mathbb{N} \\ 0^{m(n)} & \text{otherwise,} \end{cases}$$

where each $b_i \in \{0,1\}$ is a bit determined by the following procedure: (1) $x \to_{a,n(a)} (z, x^{(0)})$; (2) $G_z(x^{(i-1)}) \to_{n(a),1} (x^{(i)}, b_i)$ for each $i \in [m(n)]$. It is easily verified that G' is polynomial-time computable.

Algorithm 2: R (a nonadaptive BB reduction from distinguishing G to avoiding G')

Input : an auxiliary-input $z \in \{0,1\}^a$ and $y \in \{0,1\}^{n(a)+1}$
Oracle : $\mathcal{O}$ ($1/\gamma'$-avoiding G')

let $m := m(a + n(a))$ and select $k \sim [m]$;
let $x^{(k)} := y_{[n(a)]}$;
for $i = 1$ **to** m **do**
  if $i < k$ **then** select $\sigma_i \sim \{0,1\}$;
  else if $i = k$ **then** $\sigma_k = y_{n(a)+1}$;
  else execute $G_z(x^{(i-1)}) \to_{n(a),1} (x^{(i)}, \sigma_i)$;
query $b \leftarrow \mathcal{O}(\sigma_1 \circ \cdots \circ \sigma_m)$;
return b;

Now we define the nonadaptive reduction $R^?$ in the lemma as Algorithm 2.

We define another auxiliary-input generator $\{G''_z : \{0,1\}^{n(|z|)} \to \{0,1\}^{m(|z|+n(|z|))}\}_{z\in\{0,1\}^*}$ as $G''_z(x) := G'(z \circ x)$. If the given oracle $\mathcal{O}$ $1/\gamma'$-avoids G', then for any $z \in \{0,1\}^a$,

$$\begin{aligned}&\left|\Pr\left[\mathcal{O}(G''_z(U_{n(a)}) = 1\right] - \Pr\left[\mathcal{O}(U_{m(a+n(a))}) = 1\right]\right| \\ &\qquad = \left|\Pr\left[\mathcal{O}(G'(z \circ U_{n(a)})) = 1\right] - \Pr\left[\mathcal{O}(U_{m(a+n(a))}) = 1\right]\right| \geq \frac{1}{\gamma'(a + n(a))}.\end{aligned}$$

By the standard hybrid argument [cf. Gol01], we have that for any $z \in \{0,1\}^a$,

$$\left|\Pr\left[R^{\mathcal{O}}(z, G_z(U_{n(a)})) = 1\right] - \Pr\left[R^{\mathcal{O}}(z, U_{n(a)+1}) = 1\right]\right| \geq \frac{1}{m(a + n(a)) \cdot \gamma'(a + n(a))}.$$

By taking a polynomial γ satisfying $m(a+n(a))\cdot\gamma'(a+n(a)) \leq \gamma(n(a))$, the above inequality shows that $R^{\mathcal{O}}$ $1/\gamma$-distinguishes G for any $\mathcal{O}$ $1/\gamma'$-avoiding G'. □

Lemma 8.4.1 also implies a nonadaptive BB reduction from distinguishing AIPRG to a distributional NP problem.

Lemma 8.4.2. *For any auxiliary-input function G stretching its input and polynomial δ, there exist a language $L \in \mathsf{NP}$, a polynomial γ, and a randomized polynomial-time oracle machine $R^?$ such that for any errorless heuristic oracle $\mathcal{O}$ for (L, U) of failure probability $1/\delta$, $R^{\mathcal{O}}$ $1/\gamma$-distinguishes G. Moreover, $R^?$ is nonadaptive.*

Proof. Let $\gamma'(n) = \frac{\delta(2n)}{\delta(2n)-2}$ and $m(n) = 2n$. By Lemma 8.4.1 for G and m, there exist a polynomial γ and a nonadaptive BB reduction R_1 from $1/\gamma$-distinguishing G to $1/\gamma'$-avoiding G'.

We define the language L in the lemma as $L := \mathrm{Im}G' = \{G'(x) : x \in \{0,1\}^*\}$. Since G' is polynomial-time computable, $L \in \mathsf{NP}$.

Since δ is polynomial, there exists $n_0 \in \mathbb{N}$ such that $2^{n/2} \geq \delta(n)$ for any $n \geq n_0$. Now we construct a nonadaptive BB reduction R_2 from $1/\gamma'$-avoiding G' to an errorless heuristic algorithm for (L, U) of failure probability $1/\delta$ as Algorithm 3.

We show that R_2 is a reduction from $1/\gamma'$-avoiding G' to an errorless heuristic algorithm for (L, U) of failure probability $1/\delta$. Then by combining R_1 and R_2, we have a nonadaptive BB

Algorithm 3: R_2 (a nonadaptive BB reduction from avoiding G' to (L, U))

Input : $y \in \{0,1\}^{2n}$
Oracle : $\mathcal{O}$ (an errorless heuristic algorithm for (L, U) of failure probability $1/\delta$)

1 **if** $2n < n_0$ **then**
2 check whether $y \in \mathrm{Im}(G)$ by the brute-force search, if so, **return** 0, otherwise, **return** 1
3 **query** $b \leftarrow \mathcal{O}(y)$;
4 **if** $b \in \{1, \perp\}$ **then return** 0;
5 **else return** 1;

reduction R from $1/\gamma$-distinguishing G' to an errorless heuristic algorithm for (L, U) of failure probability $1/\delta$.

Let $y \in \{0,1\}^{2n}$ be the input for R_2. When $2n < n_0$ holds, R_2 can perfectly determine whether $y \in \mathrm{Im}G'$ and achieve the trivial threshold τ_n in the definition. Therefore, we consider only the case where $2n \geq n_0$.

Suppose that the given oracle $\mathcal{O}$ is an errorless heuristic algorithm of failure probability at most $1/\delta$, then we have that

$$y \in L\,(= \mathrm{Im}G) \Longrightarrow \mathcal{O}(y) \in \{1, \perp\}; \text{ and } \Pr_{y \sim \{0,1\}^{2n}}[\mathcal{O}(y) = \perp] \leq \delta(2n).$$

By the first implication and line 4, when y is generated by G, $R_2^{\mathcal{O}}(y)$ always outputs 0. The upper bound on the probability that $R_2^{\mathcal{O}}$ outputs 0 is given as follows:

$$\begin{aligned}
\Pr_{y \sim \{0,1\}^{2n}}[R^{\mathcal{O}}(y) = 0] &= \Pr_{y \sim \{0,1\}^{2n}}[\mathcal{O}(y) = \perp \text{ or } 1] \\
&\leq \Pr_{y \sim \{0,1\}^{2n}}[\mathcal{O}(y) = \perp] + \Pr_{y \sim \{0,1\}^{2n}}[\mathcal{O}(y) = 1] \\
&\leq \Pr_{y \sim \{0,1\}^{2n}}[\mathcal{O}(y) = \perp] + \Pr_{y}[y \in G(\{0,1\}^n)] \quad (\because \mathcal{O}(y) = 1 \Rightarrow y \in G(\{0,1\}^n)) \\
&\leq \frac{1}{\delta(2n)} + 2^{-n} \leq \frac{2}{\delta(2n)} = 1 - \frac{1}{\gamma'(n)}. \quad (\because 2n \geq n_0)
\end{aligned}$$

□

Lemma 8.4.2 implies Theorem 8.2.2 along with the following theorem shown by Bogdanov and Trevisan [BT06b].

Theorem 8.4.3 ([BT06b])**.** *For any polynomial p, any language L, and any distributional* NP *language $(L', \mathcal{D})$, if there exists a nonadaptive BB reduction from L to an errorless heuristic for $(L', \mathcal{D})$ of failure probability $1/p$, then $L \in$ coNP/poly. Moreover, if L is* NP*-hard, then* PH $= \Sigma_3^p$.

Proof of Theorem 8.2.2. By Lemma 8.4.2, there exist an NP-language L', a polynomial γ, and a nonadaptive BB reduction $R^?$ from $1/\gamma$-distinguishing G to an errorless heuristic algorithm for (L', U) of failure probability $1/n$. By combining R with the nonadaptive BB reduction in the assumption from L to $1/\gamma$-distinguishing G, we can construct a nonadaptive BB reduction from L to an errorless heuristic algorithm for (L', U) of failure probability $1/n$. Thus, by Theorem 8.4.3, the polynomial hierarchy collapses at the third[6] level. □

[6] In fact, by more careful simulation technique for HSG by Hirahara and Watanabe [HW20], we can improve the consequence on the collapse of polynomial hierarchy at the second level.

8.5 On Basing Auxiliary-Input One-Way Function on NP-Hardness

In this section, we formally show Theorem 8.2.4 based on the idea in Section 8.3.2.

Theorem (Reminder of Theorem 8.2.4)**.** *For any auxiliary-input function $f = \{f_z : \{0,1\}^n \to \{0,1\}^\ell\}_{z\in\{0,1\}^*}$ and polynomial p, if there exists a nonadaptive BB reduction from an* NP*-hard language L to $(1-1/p)$-inverting f, then* NP $\not\subseteq$ BPP *also implies that a one-way function exists (via an adaptive BB reduction).*

First, we introduce the following reduction from inverting AIOWF to a distributional NP problem, which immediately follows from Lemma 8.4.2 in Section 8.4.

Lemma 8.5.1. *For any auxiliary-input function f and reciprocals δ, δ' of polynomial, there exist an* NP*-language L and a randomized polynomial-time oracle machine $R^?$ such that for any errorless heuristic oracle $\mathcal{O}$ for (L,U) of failure probability δ', $R^{\mathcal{O}}$ $(1-\delta)$-inverting f.*

Proof. The lemma follows from Lemma 8.4.2 and the construction of auxiliary-input pseudorandom generator based on an auxiliary-input (weak) one-way function [e.g., HILL99]. □

Now we give the full proof of Theorem 8.2.4.

Proof of Theorem 8.2.4. Let $R_{L\to f}$ be the nonadaptive BB reduction from L to $(1-\delta)$-inverting f. Without loss of generality, we can assume that the failure probability of $R_{L\to f}$ is at most $1/16$ instead of $1/3$ (by taking majority vote of parallel executions) and the distributions on query are identical regardless of the query position (by adapting random permutation before asking them). We can also assume that the running time $t^{R_{L\to f}}(m)$, query complexity $q^{R_{L\to f}}(m)$, and the length of random bits $r^{R_{L\to f}}(m)$ are increasing for the input size m.

By Lemma 8.5.1, there exist an NP-language L' and a BB reduction $R_{f\to L'}$ from $(1-\delta)$-inverting f to an errorless heuristic algorithm for (L',U) of failure probability δ. Since L' is in NP, and L is NP-hard, there exists a Karp reduction $R_{L'\to L}$ from L' to L. Without loss of generality, $|R_{L'\to L}(x)| \le p(|x|)$ for some (increasing) polynomial p.

We define polynomials $q(\cdot)$, $r(\cdot)$, and $a(\cdot)$ as follows:

$$q(m) := q^{R_{L\to f}}(p(m)), \quad r(m) := r^{R_{L\to f}}(p(m)), \quad a(m) := t^{R_{L\to f}}(p(m))$$

On the execution of $R_{L\to f}(R_{L'\to L}(x))$ where $x \in \{0,1\}^m$, the number of queries, the number of random bits, and the length of queries are bounded above by $q(m)$, $r(m)$, and $a(m)$, respectively.

We also define a Turing machine $Q_m : \{0,1\}^m \times \{0,1\}^{r(m)} \to \{0,1\}^{\le a(m)}$ as $Q_m(x,s)$ outputs an auxiliary-input of the first query generated by $R_{L\to f}(R_{L'\to L}(x);s)$.

Now we construct a family of functions $g = \{g_m : \{0,1\}^{m+r(m)+n(a(m))} \to \{0,1\}^*\}_{m\in\mathbb{N}}$ by

$$g_m(x) = \left\langle z, f_z(x^f_{[n(|z|)]}) \right\rangle,$$

where $x \to_{m,r(m),n(a(m))} x^{L'} \circ s \circ x^f$ and $z = Q_m(x^{L'}, s)$.

Since f and Q_m are polynomial-time computable, g is also polynomial-time computable. We will show that if g is not one-way, then NP $\subseteq$ BPP. This immediately yields Theorem 8.2.4.

For simplicity, we consider that g_m takes as input a triple of length m, $r(m)$, and $N(m) := n(a(m))$, respectively. Suppose that g is not one-way. Then there exists a randomized polynomial-time algorithm A such that for any $m \in \mathbb{N}$,

$$\Pr_{A,U_m,U_{r(m)},U_{N(m)}}\left[A(g_m(U_m,U_{r(m)},U_{N(m)})) \notin g_m^{-1}(g_m(U_m,U_{r(m)},U_{N(m)}))\right] \le \frac{\delta(m)\cdot\delta(N(m))}{512\cdot q(m)}.$$

We also define a randomized polynomial-time algorithm A^f as

$$A^f(z,y;s_A) = \begin{cases} x^{(3)}_{[n(|z|)]} & \text{if } (x^{(1)},x^{(2)},x^{(3)}) \leftarrow A(z,y;s_A) \text{ and } z = Q_m(x^{(1)},x^{(2)}) \\ \bot & \text{otherwise.} \end{cases}$$

For any $m \in \mathbb{N}$, $x^{L'} \in \{0,1\}^m$, $s \in \{0,1\}^{r(m)}$, $x^f \in \{0,1\}^{N(m)}$, random bits $s_A \in \{0,1\}^*$ for A and A^f, and $z := Q_m(x^{L'},s)$, we have that

$$\begin{aligned}
&A(g_m(x^{L'},s,x^f);s_A) \in g_m^{-1}(g_m(x^{L'},s,x^f)) \\
&\quad\iff g_m(A(g_m(x^{L'},s,x^f);s_A)) = g_m(x^{L'},s,x^f) \ \left(= \left\langle z, f_z(x^f_{[n(|z|)]})\right\rangle\right) \\
&\quad\iff z = Q_m(x^{(1)},x^{(2)}) \text{ and } f_z(x^{(3)}_{[n(|z|)]}) = f_z(x^f_{[n(|z|)]}) \text{ where } (x^{(1)},x^{(2)},x^{(3)}) \leftarrow A(g_m(x^{L'},s,x^f);s_A) \\
&\quad\iff f_z(A^f(g_m(x^{L'},s,x^f);s_A)) = f_z(x^f_{[n(|z|)]}) \\
&\quad\iff A^f(z,f_z(x^f_{[n(|z|)]});s_A) \in f_z^{-1}(f_z(x^f_{[n(|z|)]})). \qquad (8.1)
\end{aligned}$$

Fix $m \in \mathbb{N}$ arbitrarily. Let $r := r(m)$, $N := N(m)$ and $q := q(m)$. We divide instances on L' into three sets $B_m^{L'}, G_m^{L'}$, and $N_m^{L'}$ (which stand for bad, good, and neutral, respectively) as

$$\begin{aligned}
B_m^{L'} &:= \left\{x \in \{0,1\}^m : \Pr_{A,U_r,U_N}\left[A(g_m(x,U_r,U_N)) \notin g_m^{-1}(g_m(x,U_r,U_N))\right] > \frac{\delta(N)}{216\cdot q}\right\}, \\
G_m^{L'} &:= \left\{x \in \{0,1\}^m : \Pr_{A,U_r,U_N}\left[A(g_m(x,U_r,U_N)) \notin g_m^{-1}(g_m(x,U_r,U_N))\right] \le \frac{\delta(N)}{512\cdot q}\right\}, \\
N_m^{L'} &:= \{0,1\}^m \setminus (B_m^{L'} \cup G_m^{L'}).
\end{aligned}$$

Now fix any (not-bad) instance $x \in G_m^{L'} \cup N_m^{L'}$. We also define good and bad sets on the randomness for A. Let r_A be a polynomial such that $r_A(m)$ is the number of random bits used by A for inverting g_m. Then we define the bad and good sets of randomness of A as

$$\begin{aligned}
B_{m,x}^A &:= \left\{s_A \in \{0,1\}^{r_A(m)} : \Pr_{U_r,U_N}\left[A(g_m(x,U_r,U_N;s_A)) \notin g_m^{-1}(g_m(x,U_r,U_N))\right] > \frac{\delta(N)}{16\cdot q}\right\}, \\
G_{m,x}^A &:= \{0,1\}^{r_A(m)} \setminus B_{m,x}^A.
\end{aligned}$$

For any good random bits $s_A \in G_{m,x}^A$, we define a bad set B_{m,x,s_A}^f on auxiliary-input of f as

$$B_{m,x,s_A}^f := \left\{z \in \{0,1\}^{\le a(m)} : \Pr_{U_{n(|z|)}}\left[A^f(z,f_z(U_{n(|z|)});s_A) \notin f_z^{-1}(f_z(U_{n(|z|)}))\right] > \delta(n(|z|))\right\}.$$

Consider a function $\mathcal{O}_{m,x,s_A} : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ by

$$\mathcal{O}_{m,x,s_A}(z,y) = \begin{cases} A^f(z,y;s_A) & z \in \{0,1\}^{\le a(m)} \setminus B_{m,x,s_A}^f \\ x_{z,y} & z \in B_{m,x,s_A}^f \text{ or } z \in \{0,1\}^{>a(m)}, \end{cases}$$

where $x_{z,y}$ is the lexicographically first element of $f_z^{-1}(y)$ if any, otherwise $x_{z,y} = 0$.

First we show that the above $\mathcal{O}_{m,x,s_A}(z,y)$ is indeed a $(1-\delta)$-inverting oracle for f.

Claim 8.5.2. *For any $m \in \mathbb{N}$, $x \in G_m^{L'} \cup N_m^{L'}$, and $s_A \in G_{m,x}^A$, $\mathcal{O}_{m,x,s_A}$ $(1-\delta)$-inverts f.*

Proof of Claim 8.5.2. Fix $m \in \mathbb{N}$, $x \in G_m^{L'}$, and $s_A \in G_{m,x}^A$ arbitrarily. If $z \in B_{m,x,s_A}^f \cup \{0,1\}^{>a(m)}$, by the definition of $\mathcal{O}_{m,x,s_A}$, $\mathcal{O}_{m,x,s_A}(z,y)$ must output the first inverse element of y if any.

If $z \in \{0,1\}^{\leq a(m)} \setminus B_{m,x,s_A}^f$, then we have that

$$\Pr_{x \sim \{0,1\}^{n(|z|)}}[\mathcal{O}_{m,x,s_A}(z, f_z(x)) \notin f_z^{-1}(f_z(x))] = \Pr_x[A^f(z, f_z(x); s_A) \notin f_z^{-1}(f_z(x))] \leq \delta(n(|z|)),$$

where the last inequality holds because z is not contained in B_{m,x,s_A}^f. ◇

Note that, by Claim 8.5.2, we have that for any $x' \in \{0,1\}^*$,

$$\Pr_{R_{L\to f}}[R_{L\to f}^{\mathcal{O}_{m,x,s_A}}(x') \neq L(x')] \leq 1/16. \tag{8.2}$$

If we can construct a randomized errorless heuristic algorithm B for (L', U) of failure probability at most δ, then B and $R_{f\to L'}$ yield a randomized polynomial-time algorithm $(1-\delta)$-inverting f. By using $R_{L\to f}$, we have also a randomized polynomial-time algorithm for L. Since L is NP-hard, this implies $\mathsf{NP} \subseteq \mathsf{BPP}$. Therefore, the remaining part is to construct the randomized errorless heuristic algorithm B.

Now we construct B by using A, $R_{L'\to L}$, and $R_{L\to f}$ as Algorithm 4.

Algorithm 4: B (a randomized errorless heuristic algorithm for (L', U))

Input : $x \in \{0,1\}^m$

1 *estimate the failure probability of A*
2 let $c := 0$, $M := \frac{2^{21}\cdot q^2(m)}{\delta^2(N(m))}$;
3 **repeat** M *times* **do**
4 select $s \sim \{0,1\}^{r(m)}$, $x^f \sim \{0,1\}^{N(m)}$ and compute $y = g_m(x, s, x^f)$;
5 execute $(\bar{x}^{(1)}, \bar{x}^{(2)}, \bar{x}^{(3)}) \leftarrow A(y)$;
6 **if** $g_m(\bar{x}^{(1)}, \bar{x}^{(2)}, \bar{x}^{(3)})) \neq y$ *(fail in inverting)* **then** $c := c + 1$;
7 **if** $c > M \cdot \frac{3\cdot\delta(N(m))}{1024\cdot q(m)}$ **then return** $\perp$;
8 select random bits for A^f as $s_A \sim \{0,1\}^{r_A(m)}$;
9 execute $x' \leftarrow R_{L'\to L}(x)$;
10 execute $R_{L\to f}(x')$ where for each query (z,y), answer $A^f(z,y;s_A)$;
11 if $R_{L\to f}(x')$ halts and outputs a value b, then **return** b;

We show that B is a randomized errorless algorithm for (L', U) of failure probability δ. In the subsequent argument, we use x to denote the input for B. Let $m = |x|$. By Hoeffding inequality, we can show the following claim on lines 1-7.

Claim 8.5.3. *1. If $x \in B_m^{L'}$, then $\Pr_B[B(x) = \perp] \geq 15/16$*

2. If $x \in G_m^{L'}$, then $\Pr_B[B(x) = \perp] \leq 1/16$.

Proof of Claim 8.5.3. (1) For each i-th trial in line 3, consider a Bernoulli random variable X_i which takes 1 if A fails in inverting g_m, otherwise 0. By the definition of $x \in B_m^{L'}$,

$$\mu := \mathbb{E}[X_i] > \frac{\delta(N(m))}{216 \cdot q}.$$

Therefore, we have that

$$\begin{aligned}
\Pr_B[B(x) \neq \perp] &= \Pr\left[\sum_{i=1}^{M} X_i \leq M \cdot \frac{3 \cdot \delta(N(m))}{1024 \cdot q(m)}\right] \\
&\leq \Pr\left[\frac{1}{M}\sum_{i=1}^{M} X_i - \mu \leq -\frac{\delta(N(m))}{1024 \cdot q(m)}\right] \\
&\leq \exp\left(-2 \cdot \frac{\delta^2(N(m))}{2^{20} \cdot q^2(m)} \cdot M\right) = e^{-4} < \frac{1}{16},
\end{aligned}$$

where the second inequality follows from the Hoeffding inequality.

(2) We use the same notation about X_i and μ. In the case where $x \in G_m^{L'}$, we have that

$$\mu := \mathbb{E}[X_i] \leq \frac{\delta(N(m))}{512 \cdot q}.$$

Thus, by using the Hoeffding inequality again,

$$\begin{aligned}
\Pr_B[B(x) = \perp] &= \Pr\left[\sum_{i=1}^{M} X_i > M \cdot \frac{3 \cdot \delta(N(m))}{1024 \cdot q(m)}\right] \\
&\leq \Pr\left[\frac{1}{M}\sum_{i=1}^{M} X_i - \mu \leq \frac{\delta(N(m))}{1024 \cdot q(m)}\right] \leq \exp\left(-2 \cdot \frac{\delta^2(N(m))}{2^{20} \cdot q^2(m)} \cdot M\right) < \frac{1}{16}.
\end{aligned}$$

◇

By Claim 8.5.3, we can show the following claims:

Claim 8.5.4. $\Pr_{x \sim \{0,1\}^m}[x \in B_m^{L'} \cup N_m^{L'}] \leq \delta(m)$.

Claim 8.5.5. *If* $x \in G_m^{L'}$, *then* $\Pr_B[B(x) = L'(x)] \geq 3/4$.

Claim 8.5.6. *If* $x \in N_m^{L'}$, *then* $\Pr_B[B(x) \in \{L'(x), \perp\}] \geq 3/4$.

Assume that the three claims above hold at first. Then we can show that B is a randomized errorless heuristic algorithm of failure probability δ as follows: For the condition on errorless, by Claims 8.5.3-(1), 8.5.5, and 8.5.6, for any instance $x \in \{0,1\}^m$, $B(x) \in \{L'(x), \perp\}$ with probability at least 3/4. For the condition on the failure probability, Claims 8.5.3-(1), 8.5.5, and 8.5.6 imply that $B(x)$ outputs $\perp$ with probability at least 3/4 only if $x \in B_m^{L'} \cup N_m^{L'}$. By Claim 8.5.4, the latter event occurs with probability at most $\delta(m)$ over the uniform choice of $x \in \{0,1\}^m$.

Therefore, the remaining part is only to show Claims 8.5.4, 8.5.5, and 8.5.6.

Proof of Claim 8.5.4. By the definitions of $B_m^{L'}$ and $N_m^{L'}$,

$$B_m^{L'} \cup N_m^{L'} = \left\{ x \in \{0,1\}^m : \Pr_{A,U_r,U_N}\left[A(g_m(x,U_r,U_N)) \notin g_m^{-1}(g_m(x,U_r,U_N))\right] > \frac{\delta(N)}{512 \cdot q} \right\}.$$

Remember that A satisfies

$$\Pr_{A,U_m,U_r,U_N}\left[A(g_m(x,U_r,U_N)) \notin g_m^{-1}(g_m(x,U_r,U_N))\right] \leq \delta(m) \cdot \frac{\delta(N)}{512 \cdot q}.$$

By Markov's inequality, we have that $\Pr_{x\sim\{0,1\}^m}[x \in B_m^{L'} \cup N_m^{L'}] \leq \delta(m)$. ◇

Claims 8.5.5 and 8.5.6 are immediately implied by the following Claim 8.5.7. Therefore, we first show Claims 8.5.5 and 8.5.6 by assuming Claim 8.5.7 and then show Claim 8.5.7.

Claim 8.5.7. *If $x \in G_m^{L'} \cup N_m^{L'}$, then $\Pr_B[B(x) = \neg L'(x) | B(x) \neq \perp] \leq 3/16$.*

Proof of Claim 8.5.5. If $x \in G_m^{L'}$, then

$$\begin{aligned}
\Pr_B[B(x) \neq L'(x)] &= \Pr_B[B(x) = \neg L'(x) \text{ or } B(x) = \perp] \\
&= \Pr_B[B(x) = \perp] + \Pr_B[B(x) = \neg L'(x) | B(x) \neq \perp] \\
&\leq 1/16 + 3/16 = 1/4,
\end{aligned}$$

where the last inequality follows from Claims 8.5.3-(2) and 8.5.7. ◇

Proof of Claim 8.5.6. If $x \in N_m^{L'}$, then

$$\begin{aligned}
\Pr_B[B(x) \in \{L'(x), \perp\}] &= \Pr_B[B(x) = \perp] + \Pr_B[B(x) = L'(x) | A(x) \neq \perp] \\
&\geq \Pr_B[B(x) = L'(x) | B(x) \neq \perp] \geq 13/16 > 3/4,
\end{aligned}$$

where the last inequality follows from Claim 8.5.7. ◇

Proof of Claim 8.5.7. By the assumption that $x \in G_m^{L'} \cup N_m^{L'}$, we have that

$$\Pr_{A,U_r,U_N}\left[A(g_m(x,U_r,U_N)) \notin g_m^{-1}(g_m(x,U_r,U_N))\right] \leq \frac{\delta(N)}{216 \cdot q}.$$

By Markov's inequality,

$$\Pr_{s_A\sim\{0,1\}^{r_A}}[s_A \in B_{m,x}^A] \leq \frac{1}{16}. \tag{8.3}$$

Therefore, we assume that B succeeds in selecting a good $s_A \in G_{m,x}^A$. By Claim 8.5.2, if B could simulate $\mathcal{O}_{m,x,s_A}$ in line 10 instead of $A^f(\cdot\,; s_A)$, then $R_{L\to f}$ can recognize $L'(x)$ with high probability. As shown below, however, answer by $A^f(\cdot\,; s_A)$ is consistent with answer by $\mathcal{O}_{m,x,s_A}$ with high probability over the choice of random bits for $R_{L\to f}$.

Let (z, y) be a query generated by $R_{L\to f}$. By the definition of $\mathcal{O}_{m,x,s_A}$, $A^f(z, y; s_A)$ is inconsistent with $\mathcal{O}_{m,x,s_A}(z, y)$ only if (a) $z \in B_{m,x,s_a}^f$ or (b) $|z| > a(m)$. Since $a(m)$ is the upper bound on the length of queries by $R_{L\to f}(R_{L'\to L}(x))$, the latter case (b) never occurs.

Thus, we bound above on the probability that the event (a) occurs. For the choice of the randomness $s \in \{0,1\}^{r(m)}$ to execute $R_{L\to f}$, define a bad set $B^{R_{L\to f}}_{m,x,s_A}$ as

$$B^{R_{L\to f}}_{m,x,s_A} := \left\{ s \in \{0,1\}^{r(m)} : \Pr_{U_{N(m)}}\left[A(g_m(x,s,U_{N(m)};s_A)) \notin g_m^{-1}(g_m(x,s,U_{N(m)}))\right] > \delta(N(m)) \right\}.$$

Since $s_A \in G^A_{m,x}$, we have that

$$\Pr_{U_{r(m)},U_{N(m)}}\left[A(g_m(x,U_{r(m)},U_{N(m)};s_A)) \notin g_m^{-1}(g_m(x,U_{r(m)},U_{N(m)}))\right] \le \frac{\delta(N(m))}{16\cdot q(m)}$$

By Markov's inequality,

$$\Pr_{s\sim\{0,1\}^{r(m)}}\left[s \in B^{R_{L\to f}}_{m,x,s_A}\right] \le \frac{1}{16\cdot q(m)}.$$

We define the event E_x over the choice of random bits for $R_{L\to f}$ as

$$E_x := \left(R_{L\to f}(R_{L'\to L}(x)) \text{ makes the first query } (z,y) \text{ such that } z \in B^f_{m,x,s_a}\right).$$

Then by the definitions of Q_m and g_m,

$$\begin{aligned}
\Pr_{R_{L\to f}}[E_x] &= \Pr_{s\sim\{0,1\}^{r(m)}}[Q_m(x,s) \in B^f_{m,x,s_A}] \\
&\le \Pr_{s\sim\{0,1\}^{r(m)}}\left[z \leftarrow Q_m(x,s);\ \Pr_{U_{n(|z|)}}\left[A^f(z,f_z(U_{n(|z|)});s_A) \notin f_z^{-1}(f_z(U_{n(|z|)}))\right] > \delta(n(|z|))\right] \\
&\le \Pr_{s\sim\{0,1\}^{r(m)}}\left[\Pr_{U_{N(m)}}\left[A(g_m(x,s,U_{N(m)});s_A) \notin g_m^{-1}(g_m(U_{n(m)}))\right] > \delta(N(m))\right] \quad (\because (8.1)) \\
&= \Pr_{s\sim\{0,1\}^{r(m)}}\left[s \in B^{R_{L\to f}}_{m,x,s_A}\right] \le \frac{1}{16\cdot q(m)}.
\end{aligned}$$

Since each query distribution by $R_{L\to f}$ is identical to the first query distribution, by the union bound, we have that

$$\begin{aligned}
&\Pr_{R_{L\to f}}[\text{the event (a) occurs}] \\
&\quad = \Pr_{R_{L\to f}}[R_{L\to f}(R_{L'\to L}(x)) \text{ makes at least one query } (z,y) \text{ such that } z \in B^f_{m,x,s_a}] \\
&\quad \le q(m)\cdot \Pr_{R_{L\to f}}[E_x] \le q(m)\cdot\frac{1}{16\cdot q(m)} = \frac{1}{16}.
\end{aligned}$$

Therefore, we have that

$$\Pr_{R_{L\to f}}[R^{\mathcal{O}_{m,x,s_A}}_{L\to f}(R_{L'\to L}(x)) \neq R^{A^f(\cdot;s_A)}_{L\to f}(R_{L'\to L}(x))] \le \Pr_{R_{L\to f}}[\text{the event (a) occurs}] \le 1/16. \tag{8.4}$$

Since $R_{L'\to L}$ is a Karp reduction from L' to L, $L'(x) = L(R_{L'\to L}(x))$ holds. By Eq. (8.2),

$$\Pr_{R_{L\to f}}[R^{\mathcal{O}_{m,x,s_A}}_{L\to f}(R_{L'\to L}(x)) \neq L'(x)] \le 1/16. \tag{8.5}$$

By the union bound, we conclude that (under the condition that B does not output $\bot$ in line 7)

$$\begin{aligned}\Pr_B[B(x) \neq L'(x)] &\leq \Pr_{s_A}[s_A \in B^A_{m,x}] + \Pr_{s_A, R_{L\to f}}[R^{A^f(\cdot;s_A)}_{L\to f}(R_{L'\to L}(x)) \neq L'(x) | s_A \notin B^A_{m,x}] \\ &\leq \Pr_{s_A}[s_A \in B^A_{m,x}] + \Pr_{s_A, R_{L\to f}}[R^{\mathcal{O}_{m,x,s_A}}_{L\to f}(R_{L'\to L}(x)) \neq L'(x) | s_A \notin B^A_{m,x}] \\ &\quad + \Pr_{s_A, R_{L\to f}}[R^{A^f(\cdot;s_A)}_{L\to f}(R_{L'\to L}(x)) \neq R^{\mathcal{O}_{m,x,s_A}}_{L\to f}(R_{L'\to L}(x)) | s_A \notin B^A_{m,x}] \\ &\leq \frac{1}{16} + \frac{1}{16} + \frac{1}{16} = \frac{3}{16}.\end{aligned}$$

where the last inequality follows from Eq. (8.3), (8.4), and (8.5). ◇

□

8.6 On Basing Auxiliary-Input Hitting Set Generator on NP-Hardness

In this section, we formally show Theorem 8.2.7 based on the idea in Section 8.3.3.

Theorem (Reminder of Theorem 8.2.7)**.** *Let p be a polynomial and $G := \{G_z : \{0,1\}^n \to \{0,1\}^{\ell(n)}\}_{z\in\{0,1\}^*}$ be an auxiliary-input function where $\ell(n) > (1+\epsilon)\cdot n$ for some constant $\epsilon > 0$. If there exists a nonadaptive BB reduction from an NP-hard language L to $(1-1/p)$-avoiding G, then $\mathsf{NP} \not\subseteq \mathsf{BPP}$ also implies that a one-way function exists (via an adaptive BB reduction).*

Theorem 8.2.7 obviously follows from Lemma 8.6.1 and Theorem 8.2.4.

Lemma 8.6.1. *Let δ be a reciprocal of polynomial and $G := \{G_z : \{0,1\}^n \to \{0,1\}^{\ell(n)}\}_{z\in\{0,1\}^*}$ be an auxiliary-input function where $\ell(n) > (1+\epsilon)\cdot n$ for some constant $\epsilon > 0$. If there exists a nonadaptive BB reduction from an NP-hard language L to $(1-\delta)$-avoiding G, then there exist another auxiliary-input function f and a reciprocal δ' of polynomial such that there exists a nonadaptive BB reduction from L to $(1-\delta')$-inverting f.*

To show Lemma 8.6.1, we use the following auxiliary-input analogue of Theorem 4.6.1, which holds by the same proof except the consideration of auxiliary-input (where we regard the description of circuits as auxiliary-input).

Lemma 8.6.2. *Let $\epsilon \in (0,1]$ and $\delta : \mathbb{N} \to (0,1]$ be a reciprocal of polynomial. If there exists no auxiliary-input one-way function, then for any polynomial $s(n)$, there exists a polynomial-time randomized algorithm $\mathsf{Est}_{s(n)}$ such that for any n-input circuit C of size $s(n)$,*

$$\Pr_{\mathsf{Est}_{s(n)}, x\sim\{0,1\}^n}\left[\mathsf{Est}_{s(n)}(C, C(x)) \in [(1-\epsilon)\cdot p_C(x), (1+\epsilon)\cdot p_C(x)]\right] \geq 1 - \delta(n),$$

where $p_C(x) := \Pr_{x'\sim\{0,1\}^n}[C(x') = C(x)]$.

Moreover, there exists an auxiliary-input function $f = \{f_z\}_{z\in\{0,1\}^*}$ such that $\mathsf{Est}_{s(n)}$ accesses an inverting algorithm for f nonadaptively as oracle.

Now, we show Lemma 8.6.1 to complete the proof of Theorem 8.2.7.

Proof of Lemma 8.6.1. Let $\epsilon' = \epsilon/2$, and let $R^?$ be the nonadaptive BB reduction from L to $(1-\delta)$-avoiding G. Without loss of generality, we can assume that $R^?$ makes $q(m)$ queries on input $x \in \{0,1\}^m$, where q is a polynomial, and all $q(m)$ distributions on query generated by R are identical regardless of query position by applying a random permutation before asking them.

Fix input $x \in \{0,1\}^m$ arbitrarily. Let Q_x be the distribution on the first query by $R(x)$ (which is identical to query distributions in other query positions). Let $Q_x^{(1)}$ be the distribution on auxiliary-input of Q_x.

Let $a \in \mathbb{N}$ be a length of auxiliary-input. Let $n := n(a)$ and $\ell := \ell(n)$. We divide possible queries into three sets H_x, L_x and M_x (which stand for heavy, light, and medium, respectively) as follows:

$$\begin{aligned}
H_x &:= \left\{ (z,y) \in \{0,1\}^n \times \{0,1\}^\ell : p_x(y|z) > \frac{9}{2^{(1+\epsilon')n}} \right\}, \\
L_x &:= \left\{ (z,y) \in \{0,1\}^n \times \{0,1\}^\ell : p_x(y|z) \leq \frac{1}{2^{(1+\epsilon')n}} \right\}, \\
M_x &:= \left(\{0,1\}^n \times \{0,1\}^\ell \right) \setminus (H_x \cup L_x),
\end{aligned}$$

where $p_x(y|z) = \Pr[(z,y) \sim Q_x | z \sim Q_x^{(1)}]$.

Now we define a set $\mathcal{T}_{x,a}$ of all statistical tests $T : \{0,1\}^a \times \{0,1\}^\ell \to \{0,1\}$ satisfying following conditions: for any $z \in \{0,1\}^a$,

1. $y \in \mathrm{Im}(G_z) \Longrightarrow T(z,y) = 0$
2. $(y \notin \mathrm{Im}(G_z) \ \wedge \ (z,y) \in H_x) \Longrightarrow T(z,y) = 0$
3. $(y \notin \mathrm{Im}(G_z) \ \wedge \ (z,y) \in L_x) \Longrightarrow T(z,y) = 1$

Since $\delta(n)$ is a reciprocal of polynomial, $-\log \delta(n) = O(\log n)$. Therefore, there exists $n_0 \in \mathbb{N}$ such that for any $n \geq n_0$, $n \geq \frac{1}{\epsilon'}(1 - \log \delta(n))$ holds. In the following claim, we show that each element in $\mathcal{T}_{x,a}$ avoids G_z for large enough a.

Claim 8.6.3. *For any $x \in \{0,1\}^m$ and $a \in \mathbb{N}$, if $n(a) \geq n_0$, then any $T \in \mathcal{T}_{x,a}$ $(1-\delta)$-avoids G_z for any $z \in \{0,1\}^a$.*

Proof. Fix $T \in \mathcal{T}_{x,a}$ arbitrarily. Since T satisfies the condition 1, we have that $T(z,y) = 0$ for any $z \in \{0,1\}^a$ and $y \in \mathrm{Im}(G_z)$. Thus, it is enough to show that

$$\Pr_{y \sim \{0,1\}^{\ell(n(a))}}[T(z,y) = 0] \leq \delta(n(a)).$$

Since T also satisfies the condition 3,

$$\begin{aligned}
\Pr_{y \sim \{0,1\}^\ell}[T(z,y) = 0] &\leq \Pr_{y \sim \{0,1\}^\ell}[y \in \mathrm{Im}(G_z) \ \vee \ (y,z) \notin L_x] \\
&\leq \Pr_{y \sim \{0,1\}^\ell}[y \in \mathrm{Im}(G_z)] + \Pr_{y \sim \{0,1\}^\ell}[(y,z) \in H_x \cup M_x] \\
&\leq \frac{2^n}{2^\ell} + \Pr_{y \sim \{0,1\}^\ell}[(y,z) \in H_x \cup M_x].
\end{aligned}$$

Notice that if

$$\left|\left\{y \in \{0,1\}^\ell : p_x(y|z) > 2^{-(1+\epsilon')n}\right\}\right| > 2^{(1+\epsilon')n},$$

then,

$$1 = \sum_{y\in\{0,1\}^\ell} p_x(y|z) \geq \sum_{\substack{y\in\{0,1\}^\ell:\\ (y,z)\in H_x\cup M_x}} p_x(y|z) > 2^{-(1+\epsilon')n} \cdot 2^{(1+\epsilon')n} = 1.$$

Hence, we have that

$$\left|\left\{y \in \{0,1\}^\ell : (y,z) \in H_x \cup M_x\right\}\right| = \left|\left\{y \in \{0,1\}^\ell : p_x(y|z) > 2^{-(1+\epsilon')n}\right\}\right| \leq 2^{(1+\epsilon')n}.$$

Therefore,

$$\begin{aligned}
\Pr_{y\sim\{0,1\}^\ell}[T(z,y)=0] &\leq \frac{2^n}{2^\ell} + \Pr_{y\sim\{0,1\}^\ell}[(y,z)\in H_x\cup M_x] \\
&\leq \frac{2^n}{2^\ell} + \frac{2^{(1+\epsilon')n}}{2^\ell} \\
&\leq \frac{2^n(1+2^{\epsilon' n})}{2^{(1+2\epsilon')n}} \leq \frac{2^{\epsilon' n+1}}{2^{2\epsilon' n}} = \frac{2}{2^{\epsilon' n}} \leq \delta(n(a)). \quad (\because n(a) \geq n_0)
\end{aligned}$$

◇

For $x \in \{0,1\}^m$ and a family of statistical tests $\{T_a\}_{a\in\mathbb{N}}$ where $T_a \in \mathcal{T}_{x,a}$, we define a function $\mathcal{O}_{\{T_a\}} : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ as

$$\mathcal{O}_{\{T_a\}}(z,y) = \begin{cases} 1 & \text{if } |y| \neq \ell(n(|z|)) \\ \mathbb{1}\{y \in \mathrm{Im}(G_z)\} & \text{if } |y| = \ell(n(|z|)) \ \wedge \ (2^{\epsilon' n(|z|)} \leq 144q(m) \ \vee \ n(|z|) < n_0) \\ T_{|z|}(z,y) & \text{otherwise.} \end{cases}$$

We define sets $\mathcal{T}_x$ and $\mathcal{T}$ of functions as $\mathcal{T}_x := \left\{\mathcal{O}_{\{T_a\}_{a\in\mathbb{N}}} : T_a \in \mathcal{T}_{x,a}\right\}$ and $\mathcal{T} = \bigcup_{x\in\{0,1\}^*} \mathcal{T}_x$. Then Claim 8.6.3 implies that any $\mathcal{O} \in \mathcal{T}$ $(1-\delta)$-inverts G and thus, for any $x \in \{0,1\}^*$,

$$\Pr_R[R^{\mathcal{O}}(x) \neq L(x)] \leq 1/3. \tag{8.6}$$

We can assume that $R^?(m)$ uses at most $r(m)$ random bits for any $m \in \mathbb{N}$ to create its $q(m)$ queries, where $r(\cdot)$ is a polynomial. Let $s(\cdot)$ be a polynomial satisfying that for any $m \in \mathbb{N}$ and $x \in \{0,1\}^m$, the first query by $R^?(x)$ is generated by an $s(r(m))$-size circuit which takes $r(m)$ random bits as input.

We construct a randomized polynomial-time algorithm A for L as Algorithm 5 by using the approximation algorithm $\mathsf{Est}_{s(n)}$ with $\epsilon = 1/2$ and $\delta(r(m)) = \frac{1}{32\cdot q(m)}$ in Lemma 8.6.2. Remark that A uses $\mathsf{Est}_{s(n)}$ nonadaptively (in line 3). Since $\mathsf{Est}_{s(n)}$ uses an inverting oracle for a certain auxiliary-input function f, this yields a nonadaptive BB reduction from L to inverting f.

We will show that A indeed solves L. It is not hard to see that A is polynomial-time computable and executes $\mathsf{Est}_{s(n)}$ $2q(m)$ times for the input of size m. Since the failure probability of each execution is at most $\frac{1}{32\cdot q(m)}$, the probability that at least one of the executions fails is at most $1/16$.

We assume that all executions of $\mathsf{Est}_{s(n)}$ will not fail. For $x \in \{0,1\}^*$ and $a \in \mathbb{N}$, we define a set $\mathcal{T}'_{x,a}$ composed of all statistical tests $T' : \{0,1\}^a \times \{0,1\}^{\ell(n(a))} \to \{0,1\}$ satisfying the followings: for any $z \in \{0,1\}^a$,

Algorithm 5: A (a randomized algorithm for L)

Input : $x \in \{0,1\}^m$

execute $R^?(x)$ and make $q(m)$ queries $(z_1, y_1), \ldots, (z_{q(m)}, y_{q(m)})$;
embed x to $R^?$ and create $s(r(m))$-size circuits $C_x(r)$ and $C_x^{(1)}(r)$ generating the first query and the auxiliary-input in the first query of $R^?(x; r)$, respectively;
execute $\tilde{p}_i \leftarrow \mathsf{Est}_{s(n)}(C_x, (z_i, y_i))$ and $\tilde{p}'_i \leftarrow \mathsf{Est}_{s(n)}(C_x^{(1)}, z_i)$ for each $i \in [q(m)]$;
for $i := 1$ **to** $q(m)$ **do**
    let $n_i := n(|z_i|)$;
    answer the i-th query (z_i, y_i) as follows:
        if $\exists j < i$ *such that* $(z_j, y_j) = (z_i, y_i)$ **then** return the same answer as the j-th query;
        else if $n_i < n_0$ *or* $2^{\epsilon' n_i} \leq 144q(m)$ **then** find the answer by brute-force search and return it (note that the latter condition implies $2^{n_i} \leq (144q(m))^{1/\epsilon'} \leq \mathsf{poly}(m)$);
        else if $\frac{\tilde{p}_i}{\tilde{p}'_i} \leq \frac{3}{2^{(1+\epsilon')n_i}}$ **then** return 1;
        else return 0;
if $R^?(x)$ *halts and outputs* $b \in \{0,1\}$ **then return** the same value b;

1. $(z, y) \in H_x \Longrightarrow T'(z, y) = 0$;
2. $(z, y) \in L_x \Longrightarrow T'(z, y) = 1$;

For a family of statistical tests $\{T'_a\}_{a\in\mathbb{N}}$ where $T'_a \in \mathcal{T}'_{x,a}$, we define a function $\mathcal{O}_{\{T'_a\}} : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ in the same way as $\mathcal{T}_{x,a}$. We also define the sets $\mathcal{T}'_x$ and $\mathcal{T}'$ of functions as $\mathcal{T}'_x := \left\{\mathcal{O}_{\{T'_a\}_{a\in\mathbb{N}}} : T'_a \in \mathcal{T}'_{x,a}\right\}$ and $\mathcal{T}' = \bigcup_{x\in\{0,1\}^*} \mathcal{T}'_x$.

By the correctness of $\mathsf{Est}_{s(n)}$, we have that for each $i \in [q(m)]$,

$$\frac{1}{3} \cdot p_x(y_i|z_i) \leq \frac{\frac{1}{2} \cdot \Pr[(z_i, y_i) \sim Q_x]}{\frac{3}{2} \cdot \Pr[z_i \sim Q_x^{(1)}]} \leq \frac{\tilde{p}_i}{\tilde{p}'_i} \leq \frac{\frac{3}{2} \cdot \Pr[(z_i, y_i) \sim Q_x]}{\frac{1}{2} \cdot \Pr[z_i \sim Q_x^{(1)}]} \leq 3 \cdot p_x(y_i|z_i).$$

Therefore,

$$(y_i, z_i) \in L_x \Longrightarrow \frac{\tilde{p}_i}{\tilde{p}'_i} \leq 3 \cdot p_x(y_i|z_i) \leq \frac{3}{2^{(1+\epsilon')n}},$$

and

$$(y_i, z_i) \in H_x \Longrightarrow \frac{\tilde{p}_i}{\tilde{p}'_i} \geq \frac{1}{3} \cdot p_x(y_i|z_i) > \frac{3}{2^{(1+\epsilon')n}}.$$

Hence, for any $x \in \{0,1\}^*$, $A(x)$ answers each query of $R^?(x)$ by some oracle $\mathcal{O}'$ in $\mathcal{T}'_x$ unless $\mathsf{Est}_{s(n)}$ fails. Thus, if the values of $\mathcal{O}'$ is consistent with some $\mathcal{O} \in \mathcal{T}$, then $A(x)$ can correctly simulate $(1-\delta)$-avoiding oracle for G. This motivates us to show the following claim.

Claim 8.6.4. *For any $m \in \mathbb{N}$, $x \in \{0,1\}^m$, and $\mathcal{O}' \in \mathcal{T}'_x$, there exists $\mathcal{O} \in \mathcal{T}_x$ such that*

$$\Pr_R\left[R^{\mathcal{O}'}(x) \neq R^{\mathcal{O}}(x)\right] \leq \frac{1}{16}.$$

First, we assume that Claim 8.6.4 holds. Notice that if the following three events occur on the execution of $A(x)$, then $A(x)$ outputs $L(x)$ correctly:

1. $\mathsf{Est}_{s(n)}$ does not fail, i.e., A simulates some oracle $\mathcal{O}' \in \mathcal{T}'_x$;

2. $R^{\mathcal{O}'}(x) = R^{\mathcal{O}}(x)$, where $\mathcal{O} \in \mathcal{T}_x$ is the oracle in Claim 8.6.4;

3. $R^{\mathcal{O}}(x) = L(x)$;

By Claim 8.6.4 and Eq. (8.6), the probability that each of events 1–3 does not occur is at most 1/16, 1/16, and 1/3, respectively. Therefore, for any $x \in \{0,1\}^*$,

$$\Pr_A[A(x) \neq L(x)] \leq \frac{1}{16} + \frac{1}{16} + \frac{1}{3} = \frac{11}{24}.$$

Thus, A solves L in the worst-case sense.

The remaining part is to show Claim 8.6.4.

Proof of Claim 8.6.4. Fix $x \in \{0,1\}^m$ and $\mathcal{O}' \in \mathcal{T}'_x$ arbitrarily. By the definition of $\mathcal{T}'_x$, there exists a family of statistical tests $\{T'_a\}_{a\in\mathbb{N}}$, where $T'_a \in \mathcal{T}'_{x,a}$, such that $\mathcal{O}' \equiv \mathcal{O}'_{\{T'_a\}}$.

For each $a \in \mathbb{N}$, we define a statistical test $T_a : \{0,1\}^a \times \{0,1\}^{\ell(n(a))} \to \{0,1\}$ as

$$T_a(z,y) = \begin{cases} 0 & \text{if } y \in \mathrm{Im}(G_z) \\ T'_a(z,y) & \text{otherwise.} \end{cases}$$

We also define $\mathcal{O} := \mathcal{O}_{\{T_a\}}$. It is easily verified that $T_a \in \mathcal{T}_{x,a}$. Thus, $\mathcal{O} \in \mathcal{T}_x$.

We have that

$$\Pr_R\left[R^{\mathcal{O}'}(x) \neq R^{\mathcal{O}}(x)\right] \leq \Pr_R\left[R^?(x) \text{ queries } (z,y) \text{ such that } \mathcal{O}(z,y) \neq \mathcal{O}'(z,y)\right].$$

Thus, we will bound the latter probability above by 1/16.

$$\begin{aligned}
&\mathcal{O}(z,y) \neq \mathcal{O}'(z,y) \\
&\quad\Longrightarrow 2^{\epsilon' n(|z|)} > 144q(m) \text{ and } T_{|z|}(z,y) \neq T'_{|z|}(z,y) \quad (\because \text{definitions of } \mathcal{O} \text{ and } \mathcal{O}') \\
&\quad\Longleftrightarrow 2^{\epsilon' n(|z|)} > 144q(m) \text{ and } y \in \mathrm{Im}(G_z) \text{ and } T'_{|z|}(z,y) = 1 \quad (\because \text{definitions of } T_a \text{ and } T'_a) \\
&\quad\Longrightarrow 2^{\epsilon' n(|z|)} > 144q(m) \text{ and } y \in \mathrm{Im}(G_z) \text{ and } (z,y) \notin H_x \quad (\because \text{definition of } \mathcal{T}'_x).
\end{aligned}$$

For each position $j \in [q(m)]$,

$$
\begin{aligned}
&\Pr_R\Big[R^?(x) \text{ queries } (z,y) \text{ such that } \mathcal{O}(z,y) \neq \mathcal{O}'(z,y) \text{ at the } j\text{-th query}\Big] \\
&= \Pr_{(z,y)\sim Q_x}\big[\mathcal{O}(z,y) \neq \mathcal{O}'(z,y)\big] \\
&\leq \Pr_{(z,y)\sim Q_x}\Big[2^{\epsilon' n(|z|)} > 144q(m) \text{ and } y \in \mathrm{Im}(G_z) \text{ and } (z,y) \notin H_x\Big] \\
&= \sum_{\substack{z\in\{0,1\}^*:\\ 2^{\epsilon' n(|z|)}>144q(m)}} \Pr_{(z',y)\sim Q_x}\big[y \in \mathrm{Im}(G_{z'}) \text{ and } (z',y) \in L_x \cup M_x | z' = z\big] \cdot \Pr_{z'\sim Q_x^{(1)}}[z' = z] \\
&\leq \sum_{\substack{z\in\{0,1\}^*:\\ 2^{\epsilon' n(|z|)}>144q(m)}} \sum_{\substack{y\in\mathrm{Im}(G_z):\\ (z,y)\in L_x\cup M_x}} \Pr_{(z',y')\sim Q_x}\big[y' = y | z' = z\big] \cdot \Pr_{z'\sim Q_x^{(1)}}[z' = z] \\
&= \sum_{\substack{z\in\{0,1\}^*:\\ 2^{\epsilon' n(|z|)}>144q(m)}} \sum_{\substack{y\in\mathrm{Im}(G_z):\\ (z,y)\in L_x\cup M_x}} p_x(y|z) \cdot \Pr_{Q_x^{(1)}}[z \sim Q_x^{(1)}] \\
&\leq \sum_{\substack{z\in\{0,1\}^*:\\ 2^{\epsilon' n(|z|)}>144q(m)}} \sum_{\substack{y\in\mathrm{Im}(G_z):\\ (z,y)\in L_x\cup M_x}} \frac{9}{2^{(1+\epsilon')n(|z|)}} \cdot \Pr_{Q_x^{(1)}}[z \sim Q_x^{(1)}] \quad (\because (z,y) \in L_x \cup M_x) \\
&\leq \sum_{\substack{z\in\{0,1\}^*:\\ 2^{\epsilon' n(|z|)}>144q(m)}} |\mathrm{Im}(G_z)| \cdot \frac{9}{2^{(1+\epsilon')n(|z|)}} \cdot \Pr_{Q_x^{(1)}}[z \sim Q_x^{(1)}] \\
&\leq \sum_{\substack{z\in\{0,1\}^*:\\ 2^{\epsilon' n(|z|)}>144q(m)}} \frac{9}{2^{\epsilon' n(|z|)}} \cdot \Pr_{Q_x^{(1)}}[z \sim Q_x^{(1)}] \quad (\because |\mathrm{Im}(G_z)| \leq 2^{n(|z|)}) \\
&\leq \sum_{\substack{z\in\{0,1\}^*:\\ 2^{\epsilon' n(|z|)}>144q(m)}} \frac{1}{16 \cdot q(m)} \cdot \Pr_{Q_x^{(1)}}[z \sim Q_x^{(1)}] \quad (\because 2^{\epsilon' n(|z|)} > 144q(m)) \\
&\leq \frac{1}{16 \cdot q(m)} \cdot \sum_{z\in\{0,1\}^*} \Pr_{Q_x^{(1)}}[z \sim Q_x^{(1)}] = \frac{1}{16 \cdot q(m)}.
\end{aligned}
$$

By the union bound, we conclude that

$$
\Pr_R\Big[R^?(x) \text{ queries } (z,y) \text{ such that } \mathcal{O}(z,y) \neq \mathcal{O}'(z,y)\Big] \leq \frac{q(m)}{16 \cdot q(m)} = \frac{1}{16}.
$$

$\diamond$

$\square$

Chapter 9

New and Improved Oracle Separations

In the previous chapters, we presented several characterization results related to the hardness of learning and gave new clear insights into the average-case complexity of NP and cryptography from the perspective of learning. An important future direction towards excluding Heuristica and Pessiland is to further improve our characterization results and unify the notions we discussed in this thesis.

In this chapter, we study limitations of our techniques presented in this thesis, more generally, limitations of *relativizing* proofs which hold even with additional oracle access to an *arbitrary* function $\mathcal{O}\colon \{0,1\}^* \to \{0,1\}^*$. Note that a relativizing proof is a very common and powerful framework of proofs and suffices for deriving almost all theorems previously shown in theoretical computer science by using a *reduction* with quite a few exception (e.g., $\mathsf{IP} = \mathsf{PSPACE}$ [LFKN92; Sha92] and the PCP theorem [AS98; ALMSS98]). Particularly, all the main results in Chapters 3–8 are relativized, i.e., shown by relativizing proofs.

Suppose that you try to show a conjecture that if A (e.g., $\mathsf{P} \neq \mathsf{NP}$) holds, then B (e.g., the existence of secure cryptography) also holds; however, someone shows the existence of an oracle $\mathcal{O}$ under which A holds but B does not hold (i.e., $\neg(A \to B)$). Then, as a consequence, you cannot hope any relativizing proof to show the conjecture because if such a relativizing proof exists, then $A \to B$ holds in the presence of any oracle, which contradicts the existence of $\mathcal{O}$. Namely, the oracle $\mathcal{O}$ works as a barrier against relativizing proof and is thus often called a *relativization barrier* or an *oracle separation* between the notions A and B (particularly when the conjecture is stated as $A \leftrightarrow B$, and $\neg(A \leftrightarrow B)$ holds relative to $\mathcal{O}$). One important role of an oracle separation is to avoid hopeless approach for proving the conjecture and to identify on which part we essentially need a novel idea not captured in the standard relativized framework. At a high level, for an oracle separation, we need to deeply understand and exploit the essential difference between the two discussed notions, which itself can be important knowledge for proving the conjecture. Other roles of oracle separations are explained in the survey by Fortnow [For94].

In this chapter, we present new relativization barriers against conceptually improving our main results in the previous chapters and related work. In Section 9.1, we present the oracle separation between PAC learning and the errorless average-case easiness of NP, as in Theorem 3.1.2. We also show the tight lower bound for a relativizing worst-case-to-average-case reduction within PH that matches the upper bound previously shown by Hirahara [Hir21b]. In Section 9.2, the oracle separation between PAC learning under the uniform example distribution and the error-prone average-case easiness of NP, as in Theorem 3.1.3. We remark that the main result in Section 9.2 also implies two

important results (i) the oracle separation between errorless and error-prone average-case complexity of NP, which was a longstanding open problem posed in the pioneering work by Impagliazzo [Imp95]; and (ii) a strong oracle separation between OWF and subexponentially-secure AIOWF (as Theorem 8.2.8), which improves the previous oracle separation presented in [Tre10; Nan21b]. In Section 9.3, we give a brief survey of related work on other separation results. Particularly, figure 9.1 shows a high-level overview of relativization barriers in Heuristica and Pessiland, including our results, and it can be very helpful in understanding the relationship between our work and related work.

9.1 Worst-Case vs. Average-Case Complexity

In this section, we construct "relativized Heuristica" in which there is no PAC learner with respect to almost-uniform distributions.

Theorem 9.1.1 (Theorem 3.1.2)**.** *For any arbitrary small constant $\epsilon > 0$, there exists an oracle $\mathcal{O}_\epsilon$ such that*

1. $\mathsf{DistPH}^{\mathcal{O}_\epsilon} \subseteq \mathsf{AvgP}^{\mathcal{O}_\epsilon}$*;*
2. $\mathsf{SIZE}^{\mathcal{O}_\epsilon}[n]$ *is not weakly learnable with membership queries on all uniform distributions over $S \subseteq \{0,1\}^n$ such that $|S| > 2^{(1-\epsilon)n}$ by nonuniform $2^{o(n/\log n)}$-time algorithms.*

Let us remark that Theorem 9.1.1 shows strong evidence that the learning-theoretic implication in Theorem 3.1.1 cannot be improved without a profoundly new technique. Particularly, unless we use some non-relativizing techniques, we cannot improve Theorem 3.1.1 for learning on almost-uniform example distributions even under the strong average-case assumption that $\mathsf{DistPH} \subseteq \mathsf{AvgP}$ and even with drastically weakened requirements: (a) weak learning (b) in sub-exponential time (c) with additional access to a membership query oracle.

In addition, we construct an oracle that separates the average-case complexity of PH from the worst-case complexity of $\mathsf{UP} \cap \mathsf{coUP}$ with the best possible parameters on time complexity.

Theorem 9.1.2. *There exists an oracle $\mathcal{O}$ such that*

$$\textit{(1)}\ \mathsf{DistPH}^{\mathcal{O}} \subseteq \mathsf{AvgP}^{\mathcal{O}}\ \textit{and}\ \textit{(2)}\ \mathsf{UP}^{\mathcal{O}} \cap \mathsf{coUP}^{\mathcal{O}} \not\subseteq \mathsf{SIZE}^{\mathcal{O}}[2^{o(n/\log n))}].$$

Furthermore, for all $k \in \mathbb{N}$ and constants $a > 0$, there exists an oracle $\mathcal{O}_{k,a}$ such that

$$\textit{(1)}\ \mathsf{Dist}\Sigma_k^{p\mathcal{O}_{k,a}} \subseteq \mathsf{AvgP}^{\mathcal{O}_{k,a}}\ \textit{and}\ \textit{(2)}\ \mathsf{UP}^{\mathcal{O}_{k,a}} \cap \mathsf{coUP}^{\mathcal{O}_{k,a}} \not\subseteq \mathsf{SIZE}^{\mathcal{O}_{k,a}}[2^{an/\log n}].$$

This result significantly improves the previous oracle construction of Impagliazzo [Imp11], who proved that there exist a constant $\alpha \in (0, 1/2]$ and an oracle $\mathcal{O}$ such that

$$(1)\ \mathsf{DistNP}^{\mathcal{O}} \subseteq \mathsf{AvgP}^{\mathcal{O}}\ \text{and}\ (2)\ \mathsf{UP}^{\mathcal{O}} \cap \mathsf{coUP}^{\mathcal{O}} \not\subseteq \mathsf{SIZE}^{\mathcal{O}}[2^{n^\alpha}].$$

In Theorem 9.1.2, we improve this oracle construction in the following two aspects. First, the worst-case lower bound is improved from $O(2^{n^\alpha})$ to $2^{o(n/\log n)}$. Second, the feasibility of the average-case computation is improved from DistNP to DistPH. The core concept of our improvements is to consider a switching lemma on a large alphabet, which may be of independent interest.

It is worthy of note that Hirahara [Hir21b] presented the first nontrivial worst-case-to-average-case connection for PH:

Theorem 9.1.3 ([Hir21b]). • *If* $\mathsf{DistPH} \subseteq \mathsf{AvgP}$, *then* $\mathsf{PH} \subseteq \mathsf{DTIME}[2^{O(n/\log n)}]$; *and*

• *If* $\mathsf{Dist}\Sigma^p_{k+1} \subseteq \mathsf{AvgP}$, *then* $\Sigma^p_k \subseteq \mathsf{DTIME}[2^{O(n/\log n)}]$ *for each* $k \in \mathbb{N}$.

This result is proved by a relativizing proof technique (see [HN21, Appendix A] for the details). Therefore, the time complexity $2^{n/\omega(\log n)}$ given in Theorem 9.1.2 is nearly optimal for PH and completely optimal for Σ^p_k. In this sense, we identify the capability of relativizing proof techniques for worst-case-to-average-case reductions within PH, which is the central notion in computational complexity theory.

9.1.1 Proof Idea of Oracle Separation

In this subsection, we first present our proof ideas for Theorem 9.1.2 and then for Theorem 9.1.1.

Before presenting our key idea to show Theorem 9.1.2, we first explain the idea applied in [Imp11] and the reason why it is not sufficient for the improved lower bound $2^{\Omega(n/\log n)}$.

The oracle construction by Impagliazzo [Imp11] is based on the following observation: For a hidden random function $f\colon \{0,1\}^n \to \{0,1\}^{m(n)}$, the answers to most NP computations involving in f are determined by a random restriction of the truth-table of f. Therefore, by providing access to a restrictive NP oracle $\mathcal{A}$ that answers correctly only if the random restriction determines the answer (otherwise, $\mathcal{A}$ answers $\perp$), NP problems become easy on average. By contrast, we require all the information of f to perform all NP computations involving in f. Thus, NP problems remain hard in the worst-case sense in the presence of $\mathcal{A}$. We review how this idea can be implemented.

More precisely, the oracle $\mathcal{O}$ constructed in [Imp11] consists of the following two oracles: a random permutation $\mathcal{F} = \{\mathcal{F}_n\}_{n\in\mathbb{N}}$, where $\mathcal{F}_n\colon \{0,1\}^n \to \{0,1\}^n$, and a restricted NP-oracle $\mathcal{A}$. The oracle $\mathcal{A}$ takes a nondeterministic oracle machine $M^?$, $x \in \{0,1\}^*$, and 1^{T^4}, where $T \in \mathbb{N}$, as input and simulates $M^{\mathcal{F}+\mathcal{A}}(x)$ in T steps. We remark that the simulation overhead T^4 in $\mathcal{A}$ is crucial for preventing circular calls for $\mathcal{A}$. The purpose of $\mathcal{F}$ is to make $\mathsf{UP} \cap \mathsf{coUP}$ hard by considering its inverting problem, and the purpose of $\mathcal{A}$ is to make DistNP easy on average in the relativized world. A challenging task in the construction is to preserve the worst-case hardness of NP, even in the presence of the restricted NP-oracle $\mathcal{A}$.

To satisfy this requirement, the key idea applied in [Imp11] is to let $\mathcal{A}$ reveal the values of $\mathcal{F}$ gradually according to the time bound T. The execution of a nondeterministic machine $M^{\mathcal{O}}$ is represented as a disjunctive normal form (DNF) formula in variables $F_{x,y}$ for $x, y \in \{0,1\}^*$ with $|x| = |y|$ (referred to as matching variables), which expresses the connection specified by $\mathcal{F}$ (i.e., $F_{x,y} = 1$ iff $\mathcal{F}(x) = y$). Impagliazzo's idea is to apply random restrictions to these matching variables repeatedly on the choice of $\mathcal{F}$, i.e., to determine the values of $\mathcal{F}$ in multiple steps. In the execution of $\mathcal{A}$, we determine the disclosure levels for $\mathcal{F}$ as follows. On input $(M, x, 1^{T^4})$, $\mathcal{A}$ applies only the first $i := 2^{-1}\log\log T$ restrictions to the DNF formula ϕ_M corresponding to $M^?(x)$ (where i is selected so that circular calls for $\mathcal{A}$ will not occur). If ϕ_M becomes a constant by these restrictions, then $\mathcal{A}$ returns the same constant; otherwise, $\mathcal{A}$ returns "$\perp$". We remark that, whenever $\mathcal{A}(M, x, 1^{T^4})$ returns some constant, the answer by $\mathcal{A}$ is consistent with the answer of $M^{\mathcal{O}}(x)$ executed in T steps. To solve the $\mathsf{NP}^{\mathcal{O}}$ problem $L^{\mathcal{O}}$ determined by a polynomial-time nondeterministic machine $M^{\mathcal{O}}$ on average, we query $(M, x, 1^{T^4})$ to $\mathcal{A}$ for an input x and a sufficiently large T with respect to the time bound of M and return the answer from $\mathcal{A}$. When the instance x is selected by some efficient sampler $S^{\mathcal{O}}$, $S^{\mathcal{O}}$ cannot access $\mathcal{F}$ at high disclosure levels with high probability, and the instance x is independent of such values of $\mathcal{F}$. In this case, the average-case easiness for NP follows from *the switching lemma for DNFs on matching variables*. Roughly

speaking, the lemma shows that the output of any small depth DNF formula on matching variables is fixed to a constant with high probability by applying a random restriction. Since the simulation of M by $\mathcal{A}$ is regarded as the application of a random restriction to ϕ_M, the switching lemma guarantees that the value of ϕ_M is determined with high probability, and it must be the correct answer for $L^{\mathcal{O}}$. Meanwhile, inverting $\mathcal{F}$ remains hard in the worst case as long as the inverting algorithms do not have sufficient resources to fully access $\mathcal{F}$.

The bottleneck in the above-mentioned construction lies in the bad parameters of the switching lemma for matching variables. Let N be the number of unassigned entries of $\mathcal{F}_n$ (for some $n \in \mathbb{N}$) at some stage when selecting random restrictions. To obtain the nontrivial bound on the failure probability of $\mathcal{A}$ (i.e., the probability that ϕ_M does not become a constant by a random restriction) by applying the switching lemma for matching variables, we need to additionally assign at least $N - \sqrt{N}$ entries of $\mathcal{F}_n$. To obtain the lower bound $t(n) = 2^{\Omega(n/\log n)}$ in the result, we need to apply such random restrictions $i_{\max}(n) := 2^{-1} \log \log t(n)$ times to prevent $t(n)$-time algorithms from accessing all random restrictions (i.e., full access to $\mathcal{F}$) by $\mathcal{A}$. In these settings of parameters, all the values of $\mathcal{F}$ are assigned before applying random restrictions $i_{\max}(n)$ times. In other words, $t(n)$-time algorithms can access to all the information about $\mathcal{F}$ by $\mathcal{A}$, which is sufficient to invert $\mathcal{F}$ efficiently.

Switching Lemma on General Domains

Now, we present the key idea for improving the lower bound. In this section, for simplicity, we focus on the case of separation from DistNP.

The key idea for the improvement is to apply a switching lemma *on general domains* instead of the switching lemma for matching variables, where the variables are separated into several blocks and take different alphabets in different blocks. The size of the alphabets and the probability of random restrictions also vary among the blocks. We first present the details of the switching lemma and then explain the oracle construction and the importance of large alphabets.

Let Σ be a finite set of alphabets. For a variable x that takes a value in Σ, we define a literal on x as a condition taking either of the following forms for some $a \in \Sigma$: (i) $x = a$ or (ii) $x \neq a$. Using these generalized literals, we define DNFs, conjunctive normal form formulas (CNFs), and circuits of general domains as the usual ones of a binary domain.

For $p \in [0, 1]$ and a set V of variables on Σ, we define a p-random restriction $\rho \colon V \to \Sigma \cup \{*\}$ by the following procedure. First, we select a random subset $S \subseteq V$ of size $\lfloor p|V| \rfloor$ uniformly at random. Then, we set $\rho(x) = *$ (which represents "unassigned") for $x \in S$ and assign a uniformly random value $\rho(x)$ from Σ for each $x \in V \setminus S$. For partial assignments ρ_1 to variables V_1 and ρ_2 to variables V_2, we use the notation $\rho_1\rho_2$ to represent the composite restriction to $V_1 \cup V_2$. Then, our technical lemma is stated as follows.

Lemma 9.1.4. *For $m \in \mathbb{N}$, let $\Sigma_1, \ldots, \Sigma_m$ be finite sets of alphabets, and let $V_1, \ldots, V_m$ be disjoint sets of variables, where each variable in V_i takes a value in Σ_i. For each $i \in [m]$, let ρ_i be a p_i-random restriction to V_i, where $p_i \in [0, 1]$. Then, for any t-DNF ϕ on the variables in $V_1 \cup \ldots \cup V_m$ and $k \in \mathbb{N} \cup \{0\}$, we have*

$$\Pr_{\rho_1,\ldots,\rho_m} [\phi|_{\rho_1 \ldots \rho_m} \text{ is not expressed as } k\text{-CNF }] \leq O\left(mt \cdot \max_{i \in [m]} p_i |\Sigma_i|^2\right)^{k+1}.$$

Tight Separation between UP ∩ coUP and DistNP

Here, we present the oracle construction for separating $\mathsf{UP} \cap \mathsf{coUP}$ and DistNP and explain why large alphabets on the switching lemma are important for the tight lower bound $t(n) = 2^{\Omega(n/\log n)}$.

In our construction, an oracle $\mathcal{O}$ consists of two oracles $\mathcal{V}$ and $\mathcal{A}$, where $\mathcal{V}$ makes $\mathsf{UP} \cap \mathsf{coUP}$ hard, and $\mathcal{A}$ makes DistNP easy on average in the relativized world. Further, $\mathcal{V}$ and $\mathcal{A}$ are determined by the internal random function $f = \{f_n\}_{n\in\mathbb{N}}$, where $f_n \colon \{0,1\}^n \to \Sigma_n$, and Σ_n is a subexponentially large alphabet in n. For each $n \in \mathbb{N}$, we select the random function f_n by repeatedly applying $p(n)$-random restrictions $i_{\max}(n) := \Theta(\log\log t(n))$ times and determine the disclosure levels from 1 to $i_{\max}(n)$ (i.e., full access to f_n) on the execution of $\mathcal{A}$. We define $\mathcal{V}$ as $\mathcal{V}(x,y) = 1$ if $F(x) = y$; otherwise, $\mathcal{V}(x,y) = 0$. We also define the restricted NP oracle $\mathcal{A}$ similarly to the previous construction. The easiness of DistNP follows from the switching lemma on general domains (Lemma 9.1.4).

In our oracle construction, computing f_n is hard for $t(n)$-time algorithms because any $t(n)$-time algorithm cannot obtain any information of f of the highest disclosure level from $\mathcal{A}$ by the choice of $i_{\max}(n)$. In fact, we can obtain the lower bound close to $|\Sigma_n|$ on the time complexity of computing f_n, even with access to $\mathcal{V}$. The lower bound for $\mathsf{UP}^{\mathcal{O}} \cap \mathsf{coUP}^{\mathcal{O}}$ holds because computing f_n is reducible to the following language $L^{\mathcal{O}}$ in $\mathsf{UP}^{\mathcal{O}} \cap \mathsf{coUP}^{\mathcal{O}}$:

$$L^{\mathcal{O}} = \{(x,i) : n \in \mathbb{N}, x \in \{0,1\}^n, i \in [n], \text{ and } \exists y \in \Sigma_n \text{ s.t. } f_n(x) = y \text{ and } \langle y\rangle_i = 1\},$$

where $\langle y\rangle$ denotes a (proper and unique) binary representation of $y \in \Sigma_n$.

Next, we explain the importance of large alphabets. It is natural to attempt to use a standard switching lemma on binary alphabets because the parameters achievable by such a switching lemma are significantly better than those achievable by a switching lemma on matching variables. We explain below why this approach is insufficient to obtain the tight lower bound. Let $f_n \colon \{0,1\}^n \to \{0,1\}^{\mathsf{poly}(n)}$ be a random function constructed by repeatedly applying the standard $p(n)$-random restrictions on each bit of $f_n(x)$ for every $x \in \{0,1\}^n$. Note that the output length of f_n must be at most $\mathsf{poly}(n)$ in order to make sure that $L^{\mathcal{O}} \in \mathsf{UP}^{\mathcal{O}} \cap \mathsf{coUP}^{\mathcal{O}}$. There are two conflicting requirements on the probability $p(n)$.

On one hand, to obtain the lower bound $t(n) = 2^{\Omega(n/\log n)}$, we need to let $p(n)$ be subexponentially small for the following reasons. Consider the case in which an $\mathsf{NP}^{\mathcal{O}}$ problem $L^{\mathcal{O}}$ is determined by a polynomial-time nondeterministic machine $M^{\mathcal{O}}$, and query $M^{\mathcal{O}}$ to $\mathcal{A}$ for some time bound $T = \mathsf{poly}(n)$ to solve $L^{\mathcal{O}}$ on average. For each instance $x \in \{0,1\}^n$, $M^{\mathcal{O}}(x)$ may access $f_{n'}$ by $\mathcal{A}$ for some $n' \approx t^{-1}(T)$ such that $i_{\max}(n')$ $(= \Theta(\log\log t(n')))$ is slightly larger than the disclosure level $\Theta(\log\log T)$, which is accessible to $M^?$. In this case, random restrictions for $f_{n'}$ are applied in the simulation of $M^{\mathcal{O}}(x)$ by $\mathcal{A}$. To bound the failure probability of $\mathcal{A}$ above by $1/q(n)$ for some $q(n) = \mathsf{poly}(n)$ by the switching lemma, we need to select $p(n)$ to satisfy $p(n') \approx p(t^{-1}(T)) = p(t^{-1}(\mathsf{poly}(n))) \le 1/q(n)$. To satisfy this requirement, we need to select a subexponentially small $p(n)$.

On the other hand, $p(n)$ must not be subexponentially small in order to show the worst-case lower bound on the time complexity of $L^{\mathcal{O}}$. In general, it is possible to obtain approximately 2^{d_n} as the lower bound on the time complexity of computing f_n, where d_n is the maximum number of $*$ contained in $f_n(x)$ for some $x \in \{0,1\}^n$ at the $(i_{\max}(n)-1)$ disclosure level, where remember that $i_{\max}(n) - 1 = \Theta(\log\log t(n))$. However, when we apply random restrictions for binary variables with a subexponentially small probability $p(n)$, it holds that $d_n = O(1)$ with high probability; hence, we cannot obtain the desired lower bound $2^{\Omega(n/\log n)}$ by just using a switching lemma on binary alphabets.

The switching lemma on general domains insists that the size of the alphabets only affects the probability of a random restriction multiplicatively. Thus, we can select subexponentially many alphabets even for subexponentially small $p(n)$ without affecting the failure probability. This yields sufficient $*$s in $f_n(x)$ at the $(i_{\max}(n)-1)$ disclosure level, and interestingly, it yields the tight subexponential lower bound for $\mathsf{UP} \cap \mathsf{coUP}$.

We remark that we can extend the above-mentioned argument to the case of the separation between $\mathsf{UP} \cap \mathsf{coUP}$ and DistPH by extending Lemma 9.1.4 to constant-depth circuits as the standard switching lemma.

Extending the Hardness from UP ∩ coUP to Learning

To extend the hardness result to learning, we change the oracle $\mathcal{V}$ in the above-mentioned construction to a new oracle $\mathcal{F}$ determined as follows. In addition to the random function $f = \{f_n\}_{n\in\mathbb{N}}$, where $f_n\colon \{0,1\}^n \to \Sigma_n$, we select an internal random function $g = \{g_n\}_{n\in\mathbb{N}}$, where $g_n\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ by repeatedly applying random restrictions. Intuitively, we use g as the target function $g_z(x) := g(z,x)$ for each $z \in \{0,1\}^*$ and f as the pair of locks and keys to access g through the oracle $\mathcal{F}$. Specifically, we define the oracle $\mathcal{F}$ by $\mathcal{F}(z,y,x) = g_z(x)$ if $f(z) = y$; otherwise, $\mathcal{F}(z,y,x) = 0$.

Then, we construct an oracle $\mathcal{O}$ consisting of $\mathcal{F}$ and the restricted NP oracle $\mathcal{A}$. The average-case easiness of DistNP follows from the switching lemma on general domains in a similar way, where we identify each entry $f_n(z)$ (for $z \in \{0,1\}^n$) with a variable on Σ_n and identify each entry $g(z,x)$ (for $z,x \in \{0,1\}^n$) with a binary variable.

Now, we present the proof sketch of the hardness of learning. We consider the following concept class $\mathscr{C}^{\mathcal{O}} = \{h_{z,y} : h_{z,y}(x) = \mathcal{F}(z,y,x) \text{ for } z \in \{0,1\}^n \text{ and } y \in \Sigma_n\}$. Since a worst-case learner L for $\mathscr{C}^{\mathcal{O}}$ learns $\mathcal{F}(z,y,x)$ for all $z \in \{0,1\}^n$ and $y \in \Sigma_n$, such an L must learn g_z for all $z \in \{0,1\}^n$. Note that the learner L can access $\mathcal{F}$ but not g_z through $\mathcal{F}$ unless the key $f(z)$ is identified.

We will show the upper bound on the probability that L succeeds in learning $\mathscr{C}^{\mathcal{O}}$ without sufficient resources for full access to f and g by $\mathcal{A}$. There are the following two cases for L: (1) L finds $f(z)$ for all z with notable probability, or (2) L learns g_z without identifying $f(z)$ for some z. In the former case, L essentially succeeds in computing f, which must be hard in the worst case, as discussed in the case of $\mathsf{UP} \cap \mathsf{coUP}$. In the latter case, if we consider the case of learning g_z on the uniform distribution over unrevealed entries of g_z at the $(i_{\max}(n)-1)$ disclosure level, then L cannot distinguish the value of g_z with a truly random value on the support of the example distribution even with access to $\mathcal{A}$. Thus, L cannot learn g_z even weakly on such an example distribution. In fact, we can show that there exists an index z with high probability such that the value $f(z)$ is unassigned and many $*$s remain in the truth table of g_z at the $(i_{\max}(n)-1)$ disclosure level. This yields the subexponential lower bound of weak learning on almost-uniform distributions.

Based on the ideas above, we prove Theorems 9.1.1 and 9.1.2 in subsequent sections. In Section 9.1.2, we show the switching lemma on general domains. In Section 9.1.3 and 9.1.4, we prove Theorems 9.1.2 and 9.1.1.

9.1.2 Switching Lemma on General Domains

In this section, we extend the switching lemma of a binary domain to general domains. Our proof mainly follows the proof presented by Razborov [Raz93].

We remark that, for $p \in [0,1]$ and a set V of variables on alphabets Σ, we define a p-random restriction $\rho\colon V \to \Sigma \cup \{*\}$ by the following procedure. First, we select a random subset $S \subseteq V$ of size $\lfloor p|V| \rfloor$ uniformly at random. Then, we set $\rho(x) = *$ for $x \in S$ and assign a random value $\rho(x) \sim \Sigma$ for each $x \in V \setminus S$. For every restriction ρ to S and every function f defined on S, we let $f|_\rho$ denote the restricted function obtained by applying a partial assignment to f according to ρ.

Lemma 9.1.5 (Lemma 9.1.4)**.** *For $m \in \mathbb{N}$, let $\Sigma_1, \ldots, \Sigma_m$ be finite sets of alphabets, and let $V_1, \ldots, V_m$ be disjoint sets of variables, where each variable in V_i takes a value in Σ_i. For each $i \in [m]$, let ρ_i be a p_i-random restriction to V_i, where $p_i \in [0,1]$. Then, for any t-DNF ϕ on the variables in $V_1 \cup \ldots \cup V_m$ and $k \in \mathbb{N} \cup \{0\}$, we have*

$$\Pr_{\rho_1,\ldots,\rho_m}[\phi|_{\rho_1 \ldots \rho_m} \text{ is not expressed as } k\text{-CNF }] \leq O\left(mt \cdot \max_{i \in [m]} p_i |\Sigma_i|^2\right)^{k+1}.$$

Proof. Each literal in ϕ of the form $(x \neq a)$ (for some $a \in \Sigma_i$) is expressed as $\bigvee_{b \in \Sigma_i : b \neq a}(x = b)$. Note that if we apply this transformation to all literals of the form $(x \neq a)$ in ϕ and expand them to obtain a DNF formula, these operations do not change the width of the original DNF ϕ. Thus, without loss of generality, we can assume that ϕ does not contain any literal of the form $(x \neq a)$.

For each $i \in [m]$, let $M_i = |\Sigma_i|$, $N_i = |V_i|$, and $n_i = \lfloor p_i N_i \rfloor$. To prove the lemma, we assume that $\phi|_{\rho_1 \ldots \rho_m}$ is not expressed as k-CNF and show that $\rho = \rho_1 \cdots \rho_m$ has a short description for estimating the number of such restrictions.

We can select a partial assignment π to $V_1 \cup \ldots \cup V_m$ of size at least $k+1$ such that $\phi|_{\rho\pi} \equiv 0$, but for any proper subrestriction π' of π, $\phi|_{\rho\pi'} \not\equiv 0$ (otherwise, $\phi|_\rho$ must be expressed as k-CNF). We also select subrestrictions π_j of π and restrictions σ_j inductively on $j \leq s\,(\leq k+1)$ by the following procedure. Assume that $(\pi_1, \sigma_1), \ldots, (\pi_{i-1}, \sigma_{j-1})$ have been determined, and $\pi_1 \cdots \pi_{j-1} \not\equiv \pi$; if not, we complete the procedure. Since $\pi_1 \cdots \pi_{j-1}$ is a proper subrestriction of π, we have $\phi|_{\rho\pi_1 \cdots \pi_{j-1}} \not\equiv 0$, and we can select the first term τ_j (in some fixed order) such that the value of τ_j is not determined by $\rho\pi_1 \cdots \pi_{j-1}$. Since $\tau_j|_{\rho\pi} \equiv 0$, there must exist a set S_j of variables that are contained in τ_j, unassigned by $\pi_1 \cdots \pi_{j-1}$ but assigned by π. We define σ_j as a partial assignment to S_j, which is consistent with the literals in τ_j. We also define π_j as the corresponding subrestriction of π to S_j. This procedure is repeated until $\pi_1 \cdots \pi_j \equiv \pi$ holds; let s denote the index j at the end. For convenience, we trim S_s (and π_s, σ_s correspondingly) in some arbitrary manner to satisfy $k+1 = |S_1 \cup \cdots \cup S_s|$.

For each $j \in [s]$, let P_j denote the set of indices in $[t]$, which indicates the position of the variables in S_i among the literals in τ_j, and let Q_j denote $Q_j = (\pi_j(v_1), \ldots, \pi_j(v_{|P_j|}))$, where $v_{j'}$ is the j'-th variable indicated by P_j. For each $i \in [m]$, let k_i be the number of variables in V_i that are assigned by $\sigma_1 \ldots \sigma_s$, i.e., we have $k+1 = \sum_i k_i$.

We claim that ρ can be reconstructed from the composite restriction $\rho' = \rho\sigma_1 \cdots \sigma_s$, $P_1, \ldots, P_s$, and $Q_1, \ldots, Q_s$ by the following procedure: (0) let $j = 1$; (1) find the first term not to become 0 by ρ', which must be τ_j by the construction; (2) obtain σ_j and π_j from ρ', P_j, and Q_j; (3) let $\rho' := \rho\pi_1 \ldots \pi_j \sigma_{j+1} \ldots \sigma_m$ and $j := j+1$, and repeat (1) and (2) to obtain σ_j and π_j; (4) repeat (3) until all of $\sigma_1, \ldots, \sigma_s$ are obtained; then, ρ can be reconstructed from ρ' and $\sigma_1, \ldots, \sigma_s$.

Therefore, ρ is represented by $P_1, \ldots, P_s, Q_1, \ldots, Q_s$, and the composite restriction ρ' that has $(n_i - k_i)$ $*$s on V_i for each $i \in [m]$. For each choice of $k_1, \ldots, k_m$ such that $k+1 = \sum_i k_i$ and each $i \in [m]$, the possible choice of P_i is at most t^{k_i}, and the possible choice of Q_i is at most $M_i^{k_i}$. Thus,

the possible number of such expressions is at most

$$C\cdot \sum_{k_i:k+1=\sum_i k_i} \prod_{i\in[m]} \binom{N_i}{n_i-k_i}\cdot M_i^{N_i-n_i+k_i}\cdot t^{k_i}\cdot M_i^{k_i} = C\cdot \sum_{k_i:k+1=\sum_i k_i} t^{k+1}\cdot \prod_{i\in[m]} \binom{N_i}{n_i-k_i}\cdot M_i^{N_i-n_i+k_i}\cdot M_i^{k_i},$$

for some absolute constant C.

If $\max_{i\in[m]} n_i/N_i \geq 1/2$, then the lemma holds trivially because $2\max_{i\in[m]} p_i \geq 1$. Therefore, we can assume that $n_i/N_i < 1/2$, i.e., $n_i < N_i/2$ for each $i \in [m]$. Then, we can establish the upper bound on the probability as follows:

$$\begin{aligned}
\Pr_{\rho_1,\ldots,\rho_m}[\phi|_{\rho_1\ldots\rho_m} \text{ is not expressed as } k\text{-CNF }] &\leq C\cdot \sum_{k_i:k+1=\sum_i k_i} t^{k+1}\cdot \prod_{i\in[m]} \frac{\binom{N_i}{n_i-k_i}\cdot M_i^{N_i-n_i+2k_i}}{\binom{N_i}{n_i}\cdot M_i^{N_i-n_i}} \\
&\leq C\cdot \sum_{k_i:k+1=\sum_i k_i} t^{k+1}\cdot \prod_{i\in[m]} \frac{n_i^{k_i}}{(N_i-n_i)^{k_i}} M_i^{2k_i} \\
&\leq C\cdot \sum_{k_i:k+1=\sum_i k_i} t^{k+1}\cdot \max_{i\in[m]} \left(\frac{n_i M_i^2}{N_i-n_i}\right)^{k+1} \\
&\leq C\cdot (mt)^{k+1}\cdot \max_{i\in[m]} \left(\frac{n_i M_i^2}{N_i-n_i}\right)^{k+1} \\
&\leq C\cdot (mt)^{k+1}\cdot \max_{i\in[m]} \left(\frac{2n_i M_i^2}{N_i}\right)^{k+1} \\
&= O\left(mt\cdot \max_{i\in[m]} \frac{n_i M_i^2}{N_i}\right)^{k+1} \\
&= O\left(mt\cdot \max_{i\in[m]} p_i|\Sigma_i|^2\right)^{k+1}.
\end{aligned}$$

□

The above-mentioned lemma implies the following by considering the negation of a given CNF formula.

Lemma 9.1.6. *For $m \in \mathbb{N}$, let $\Sigma_1,\ldots,\Sigma_m$ be finite sets of alphabets, and let $V_1,\ldots,V_m$ be disjoint sets of variables, where each variable in V_i takes a value in Σ_i. For each $i \in [m]$, let ρ_i be a p_i-random restriction to V_i, where $p_i \in [0,1]$. Then, for any t-CNF ϕ on the variables in $V_1 \cup \ldots \cup V_m$ and $k \in \mathbb{N} \cup \{0\}$, we have*

$$\Pr_{\rho_1,\ldots,\rho_m}[\phi|_{\rho_1\ldots\rho_m} \textit{ is not expressed as } k\textit{-DNF }] \leq O\left(mt\cdot \max_{i\in[m]} p_i|\Sigma_i|^2\right)^{k+1}.$$

Now, we extend the above-mentioned results to constant-depth circuits on general domains. For any depth-d circuit, we number each layer from 0 (bottom) to d (top), where layer 0 consists of input gates and layer d consists of the topmost $\vee$- or $\wedge$-gate. Without loss of generality, we can assume that each depth-d circuit satisfies the following properties: (1) each input gate is a literal

taking the form of either $(x = a)$ or $(x \neq a)$ for some alphabet a; (2) each layer (from 1 to d) contains either $\vee$-gates or $\wedge$-gates; and (3) the type of gate (i.e., $\vee$ or $\wedge$) alternates at adjacent layers. For any depth-d circuit, we define its width by the maximum number of literals (i.e., input gates) that are connected to the same gate and define its internal size by the total number of gates at layers $2, 3, \ldots d$ (note that we do not contain the number of gates at layer 1). Then, our technical lemma is stated as follows.

Lemma 9.1.7. *For $m \in \mathbb{N}$, let $\Sigma_1, \ldots, \Sigma_m$ be finite sets of alphabets, and let $V_1, \ldots, V_m$ be disjoint sets of variables, where each variable in V_i takes a value in Σ_i. For each $i \in [m]$, let ρ_i be a p_i-random restriction to V_i, where $p_i \in [0, 1]$. Then, for any depth-d circuit C on the variables in $V_1 \cup \ldots \cup V_m$ of width $\leq t$ and internal size $\leq c2^t$ (for some constant c), we have*

$$\Pr_{\rho_1,\ldots,\rho_m}[C|_{\rho_1 \ldots \rho_m} \text{ is not a constant }] \leq O\left(mt \cdot \max_{i\in[m]} p_i^{\frac{1}{d}} |\Sigma_i|^2\right).$$

Proof. For each $i \in [d]$, let s_i be the number of gates at layer i.

First, we consider only the case where the following holds: for all $i \in [m]$,

$$\lfloor p_i N \rfloor \leq \lfloor p_i^{1/d} \lfloor p_i^{1/d} \cdots \lfloor p_i^{1/d} N \underbrace{\rfloor \cdots \rfloor}_{d-1 \text{ times}} \rfloor. \tag{9.1}$$

In this case, we can regard a p_i-restriction ρ_i as consecutive applications of $p_i^{1/d}$-random restrictions $\rho_i^{(1)}, \ldots, \rho_i^{(d-1)}$, and one remaining random restriction. For each $i \in [d-1]$, let $\rho^{(i)} \equiv \rho_1^{(i)} \ldots \rho_m^{(i)}$.

We assume that layer 1 consists of $\wedge$-gates (in the case of $\vee$-gates, we can show the lemma in the same manner). In this case, each gate in layer 2 is regarded as t-DNF; thus, we can apply Lemma 9.1.5 and show that all the gates at layer 2 are transformed into t-CNF with a probability of at least $1 - s_2 \cdot (c'mt \cdot \max_{i\in[m]} p_i^{1/d}|\Sigma_i|^2)^t$ for some absolute constant c'. If such an event occurs, then each $\vee$-gate at layer 2 collapses into its parent node. Thus, the depth decreases by 1. Since the resulting depth-$(d-1)$ circuit has width t, we can apply Lemma 9.1.6 and the same argument at layer 3. We repeat the same argument $(d-2)$ times for $\rho_i^{(1)}, \ldots, \rho_i^{(d-2)}$ at layers $2, \ldots, d$, respectively. Then, the resulting circuit becomes a depth-2 circuit of width t (i.e., t-DNF or t-CNF) with a probability of at least

$$1 - (s_2 + s_3 + \ldots + s_d) \cdot (c'mt \cdot \max_{i\in[m]} p_i^{1/d}|\Sigma_i|^2)^{t+1} \geq 1 - (c2^t) \cdot (c'mt \cdot \max_{i\in[m]} p_i^{1/d}|\Sigma_i|^2)^{t+1}.$$

We apply Lemmas 9.1.5 and 9.1.6 and show that the resulting circuit becomes a constant by $\rho^{(d-1)}$ with a probability of at least $1 - c'mt \cdot \max_{i\in[m]} p_i^{1/d}|\Sigma_i|^2$. Without loss of generality, we can assume that $2c'mt \cdot \max_{i\in[m]} p_i^{1/d}|\Sigma_i|^2 < 1$; otherwise, the lemma holds trivially. Thus, we conclude that

$$\begin{aligned}
\Pr_{\rho_1,\ldots,\rho_m}[C|_{\rho_1 \ldots \rho_m} \text{ is not a constant }] &\leq \Pr_{\rho^{(1)},\ldots,\rho^{(d-1)}}\left[C|_{\rho^{(1)}\ldots\rho^{(d-1)}} \text{ is not a constant }\right] \\
&\leq c(2c'mt \cdot \max_{i\in[m]} p_i^{1/d}|\Sigma_i|^2)^{t+1} + c'mt \cdot \max_{i\in[m]} p_i^{1/d}|\Sigma_i|^2 \\
&\leq 2cc'mt \cdot \max_{i\in[m]} p_i^{1/d}|\Sigma_i|^2 + c'mt \cdot \max_{i\in[m]} p_i^{1/d}|\Sigma_i|^2 \\
&= O\left(mt \cdot \max_{i\in[m]} p_i^{\frac{1}{d}}|\Sigma_i|^2\right).
\end{aligned}$$

Next, we consider the case where (9.1) does not hold for some $i \in [m]$. We can assume that $p_i^{1/d} < 1/4$; otherwise, we have $1/4 \leq p_i^{1/d} \leq \max_{i\in[m]} p_i^{1/d}|\Sigma_i|^2$, and the lemma holds trivially. In this case, we can show that

$$\begin{aligned}
p_i N &\geq \lfloor p_i N \rfloor \\
&> \lfloor \lfloor \cdots \lfloor p_i^{1/d} N \rfloor \cdots \rfloor \rfloor \\
&\geq \lfloor \lfloor \cdots \lfloor p_i^{1/d} (p_i^{1/d} N - 1) \rfloor \cdots \rfloor \rfloor \\
&\geq p_i^{(d-1)/d} N - (1 + p_i^{1/d} + p_i^{2/d} + \cdots + p_i^{(d-2)/d}) \\
&\geq p_i^{(d-1)/d} N - 2.
\end{aligned}$$

By rearranging the above, we have

$$\frac{2}{p_i^{(d-1)/d} N} \geq 1 - p_i^{1/d} > \frac{3}{4},$$

and

$$p_i N = p_i^{1/d} \cdot p_i^{(d-1)/d} N < \frac{1}{4} \cdot \frac{8}{3} = \frac{2}{3}.$$

Therefore, $\lfloor p_i N \rfloor = 0$ holds, and all the variables in V_i are fully determined by ρ_i. Thus, we can ignore such i in the argument above. □

9.1.3 Oracle Separation: UP ∩ coUP and Distributional PH

Now, we improve the oracle separation in [Imp11] by applying Lemma 9.1.7. In this section, we present the following theorem (i.e., the first item of Theorem 9.1.2). Note that the second item of Theorem 9.1.2 is shown in a similar way by changing the parameters (we will discuss this at the end of this section).

Theorem 9.1.8. *For any function $\epsilon(n)$ such that $\omega(1) \leq \epsilon(n) \leq n/\omega(\log^2 n)$, there exists an oracle $\mathcal{O}_\epsilon$ satisfying (1) $\mathsf{DistPH}^{\mathcal{O}_\epsilon} \subseteq \mathsf{AvgP}^{\mathcal{O}_\epsilon}$ and (2) $\mathsf{UP}^{\mathcal{O}_\epsilon} \cap \mathsf{coUP}^{\mathcal{O}_\epsilon} \not\subseteq \mathsf{SIZE}^{\mathcal{O}_\epsilon}[2^{O(\frac{n}{\epsilon(n)\log n})}]$.*

Construction of Random Oracle

Let $\epsilon : \mathbb{N} \to \mathbb{N}$ denote a parameter such that $\omega(1) \leq \epsilon(n) \leq n/\omega(\log^2 n)$.

Construction. *$\mathcal{O}_\epsilon = \mathcal{V} + \mathcal{A}$, where each oracle is randomly selected by the following procedure:*

1. *Define functions t, p, ℓ, and $i_{\max}$ as*

$$t(n) = 2^{\frac{n}{\epsilon(n)\cdot\log n}},\ p(n) = t(n)^{-\epsilon(n)^{1/2}},\ \ell(n) = t(n)^2,\ \text{and}\ i_{\max}(n) = \log\log t(n).$$

2. *For each $n \in \mathbb{N}$, define a set $V_{n,0}$ of variables on alphabet Σ_n of size $\ell(n)$ as*

$$V_{n,0} = \{F_x : x \in \{0,1\}^n\}.$$

We assume that each alphabet in Σ_n has a binary representation of length at most $\lceil \log \ell(n) \rceil$.

3. *For each $n \in \mathbb{N}$ and $i \in [i_{\max}(n) - 1]$, we inductively (on i) define a $p(n)$-random restriction $\rho_{n,i}$ to $V_{n,i-1}$ and define a subset $V_{n,i} \subset V_{n,i-1}$ of variables as*

$$V_{n,i} = \{v \in V_{n,i-1} : \rho_{n,i}(v) = *\}.$$

We also define $\rho_{n,i_{\max}(n)}$ as a 0-random restriction (i.e., a full assignment) to $V_{n,i_{\max}(n)-1}$. Let $\rho_{n,i} \equiv \rho_{n,i_{\max}(n)}$ for $i \geq i_{\max}(n) + 1$. For simplicity, we may identify $\rho_{n,i}$ with the composite restriction $\rho_{n,1} \dots \rho_{n,i}$ to $V_{n,0}$ for each n and i.

4. *Let $f = \{f_n\}_{n\in\mathbb{N}}$, where $f_n \colon \{0,1\}^n \to \Sigma_n$ is a random function defined as $f_n(x) = \rho_{n,i_{\max}(n)}(F_x)$.*

5. *Define $\mathcal{V}$ as follows:*

$$\mathcal{V}(x,y) = \begin{cases} 1 & \text{if } y = f(x) \\ 0 & \text{otherwise.} \end{cases}$$

6. *Define $\mathcal{A}$ as follows: On input $(\langle M, d\rangle, x, 1^{T^2})$, where M is an oracle machine, $d \in \mathbb{N}$, $x \in \{0,1\}^*$, and $T \in \mathbb{N}$, the oracle $\mathcal{A}$ returns the value in $\{0, 1, \perp\}$ determined according to the following procedure:*

   1: Let $i := \log\log T$.
   2: Construct a depth $d+2$ circuit C corresponding to the quantified formula

$$\exists w_1 \in \{0,1\}^{|x|} \forall w_2 \in \{0,1\}^{|x|}, \dots, Q_d w_d \in \{0,1\}^{|x|}, M^{\mathcal{O}}(x, w_1, w_2, \dots, w_d),$$

   where $Q_d = \exists$ if d is an odd number; otherwise, $Q_d = \forall$.
   First, we construct a depth d circuit that represents the above-mentioned quantified formula, where each leaf corresponds to $M^{\mathcal{O}}(x, w_1, w_2, \dots, w_d)$ for some $w_1, w_2, \dots, w_d$, where we truncate $w_1, w_2, \dots, w_d$ into a string of length T because we will execute M in only T steps. Then, we replace each leaf with a DNF formula of width T to obtain the circuit C, where each term corresponds to one possible choice of $\mathcal{V}$ such that $M^{\mathcal{V}+\mathcal{A}}(x, w_1, w_2, \dots, w_d)$ halts with an accepting state after execution in T steps. In other words, we consider each function $f' = \{f'_n\}_{n=1}^T$, where $f'_n \colon \{0,1\}^n \to \Sigma_n$, define an oracle $\mathcal{V}'$ in the same manner as $\mathcal{V}$, and execute $M^{\mathcal{V}'+\mathcal{A}}(x, w_1, w_2, \dots, w_d)$ in T steps. If M queries (x, y) to $\mathcal{V}'$, and the answer is 1 (resp. 0), then we add a literal $(F_x = y)$ (resp. $(F_x \neq y)$) to the corresponding term. Finally, we construct a circuit $C = \bigvee_{f'} C_{f'}$ on $V_{1,0}, \dots, V_{T,0}$.
   By the construction described above, the above-mentioned quantified formula is satisfied with the execution of $M^{\mathcal{V}+\mathcal{A}}$ in T steps iff C returns 1 when it is restricted by $\rho_{1,i_{\max}(1)}, \dots, \rho_{T,i_{\max}(T)}$. We can also easily verify that the width of C is at most T, and the internal size of C is at most 2^{T+1}.
   *3: If $C|_{\rho_{1,i},\dots,\rho_{T,i}} \equiv b$ for some $b \in \{0,1\}$, then **return** b; otherwise, **return** "$\perp$".*

To verify that the above-mentioned $\mathcal{A}$ is not circular on recursive calls for $\mathcal{A}$, it is sufficient to show the following.

Lemma 9.1.9. *For each input, the value of $\mathcal{A}(\langle M, d\rangle, x, 1^{T^2})$ is determined only by $\rho_{n,j}$ for $n \leq T$ and $j \leq \log\log T$ $(= i)$.*

Proof. We show the lemma by induction on T. We consider the execution of $(\langle M, d\rangle, x, 1^{T^2})$. We remark that $\mathcal{A}$ first makes a depth $d+2$ circuit C based on M, and C is independent of the value of $\mathcal{V}$.

We assume that M makes some valid query $(\langle M', d'\rangle, x', 1^{T'^2})$ to $\mathcal{A}$ recursively on constructing C. Since the length of such a query is at most T, we have $T'^2 \le T$. If we let $i' = \log\log T'$, then we have

$$i' = \log\log T' \le \log\log T^{\frac{1}{2}} = \log\log T - 1.$$

By the induction hypothesis, the recursive answer of $\mathcal{A}$ is determined by only $\rho_{n,j}$ for $n \le T'$ and $j \le i-1$, and so is C. The lemma holds because the answer of $\mathcal{A}$ is determined by restricting C by $\rho_{n,j}$ for $n \le T$ and $j \le i$. □

Lemma 9.1.10. *For $\epsilon(n) = \omega(1)$, it holds that $|V_{n,i_{\max}(n)-1}| = 2^{\Omega(n)}$ for sufficiently large n.*

Proof. Since $\epsilon(n) = \omega(1)$, we have $t(n) \le 2^n$ and $i_{\max}(n) \le \log n$ for sufficiently large n. Thus, for sufficiently large n, we have

$$p(n)^{i_{\max}(n)-1} \ge t(n)^{-\epsilon(n)^{1/2}\log n} = 2^{-\frac{n\log n}{\epsilon(n)^{1/2}\log n}} \ge 2^{-\frac{n}{\omega(1)}},$$

and $|V_{n,i_{\max}(n)}| = \Omega(p(n)^{i_{\max}(n)-1}\cdot 2^n) = 2^{\Omega(n)}$. □

Note that we may omit the subscript ϵ from $\mathcal{O}_\epsilon$.

Worst-Case Hardness of UP ∩ coUP

We introduce the following useful lemma to show the worst-case hardness.

Lemma 9.1.11 ([GT00])**.** *Let $S, T \subseteq \{0,1\}^*$ be finite subsets of the same size N, and let $b\colon S \to T$ be a bijection. Let $A^?$ be a deterministic oracle machine that makes at most q queries to b. If $A^b(y) = b^{-1}(y)$ for all $y \in T$, then b has the representation of length at most $2\log\binom{N}{a} + \log((N-a)!)$ when A is given, where $a = N/(q+1)$.*

Now, we show the subexponential worst-case hardness of computing f.

Theorem 9.1.12. *For any function ϵ such that $\omega(1) \le \epsilon(n) \le n/\omega(\log^2 n)$, with probability 1 over the choice of $\mathcal{O}_\epsilon$, no nonuniform oracle machine can compute f within $t(n) = 2^{\frac{n}{\epsilon(n)\cdot\log n}}$ steps, where f is the random function selected in $\mathcal{O}_\epsilon$.*

Proof. First, we assume that for any nonuniform oracle machine A and any sufficiently large input size $n \in \mathbb{N}$,

$$P_{A,n} := \Pr_{\mathcal{O}}\left[\forall x \in \{0,1\}^n, A^{\mathcal{O}}(x) \text{ outputs } f(x) \text{ within } t(n) \text{ steps}\right] \le 2^{-\Omega(t(n)^2)}. \tag{9.2}$$

Since every nonuniform $t(n)$-time oracle machine is described by $O(t(n))$ bits, by the union bound, we have that for any sufficiently large $n \in \mathbb{N}$, no nonuniform $t(n)$-time oracle machine satisfies the event above with probability at least $1 - 2^{O(t(n))}\cdot 2^{-\Omega(t(n)^2)} = 1 - 2^{-\Omega(t(n)^2)} = 1 - n^{-\omega(1)}$ over the choice of $\mathcal{O}$, where we use the assumption that $\epsilon(n) \le n/\omega(\log^2 n)$. By the Borel–Cantelli lemma, the same event holds for all sufficiently large $n \in \mathbb{N}$, i.e., no nonuniform $t(n)$-time oracle machine can compute f, with probability 1 over the choice of $\mathcal{O}$.

In the remainder of the proof, we show Eq. (9.2). Fix sufficiently large $n \in \mathbb{N}$ and a nonuniform $t(n)$-time oracle machine A arbitrarily so that $N := |V_{n,i_{\max}(n)-1}| = 2^{\Omega(n)}$ as in Lemma 9.1.10.

First, we fix random restrictions $\rho_{n',i}$ except for $\rho_{n,i_{\max}(n)}$ arbitrarily and use the notation ρ to denote the restriction. We remark that ρ determines $V_{n,i_{\max}(n)-1}$. Even under the condition on ρ, the value of $f_n(x)$ for each x such that $F_x \in V_{n,i_{\max}(n)-1}$ is selected from Σ_n uniformly at random (by $\rho_{n,i_{\max}(n)}$). For any choice of ρ, we can divide a random selection of $\rho_{n,i_{\max}(n)}$ into the following two steps without loss of generality: (i) select N elements $y_1, \dots, y_N \sim \Sigma_n$ uniformly at random, and (ii) select a random bijection $b\colon V_{n,i_{\max}(n)-1} \to [N]$ to assign each value of $f(x)$ as $f(x) \equiv y_{b(x)}$ for each $x \in V_{n,i_{\max}(n)-1}$.

We consider an arbitrary choice of $\mathcal{O}$ except for the aforementioned bijection b and use the notation C to refer to such a partial choice of $\mathcal{O}$. We regard C as a condition on the choice of $\mathcal{O}$. We say that the partial choice C is bad if there are two distinct indices $j_1, j_2 \in [N]$ such that $y_{j_1} = y_{j_2}$. Since $y_1, \dots, y_N$ are uniformly and independently selected from $|\Sigma_n| = \ell(n)$ elements, by the union bound, we obtain that

$$\Pr_{\mathcal{O}}[C \text{ is bad}] \le N^2 \cdot 2^{-\ell(n)} \le 2^{2n} \cdot 2^{-\ell(n)} = 2^{-\Omega(t(n)^2)}.$$

Below, we assume the event that C is not bad. Under this condition, we can identify the random bijection b with a random bijection b' that maps $\{y_1, \dots, y_N\}$ to $V_{n,i_{\max}(n)-1}$.

Since we execute A in $t(n)$ steps, the length of the query made by A is at most $t(n)$. Thus, A can only access $\mathcal{A}(M, x, 1^{T^2})$ for $T \le t(n)^{1/2}$. For such T, we have

$$i = \log\log T \le \log\log t(n) - 1 = i_{\max}(n) - 1.$$

Therefore, by Lemma 9.1.9, the answers of $\mathcal{A}$ to queries made by A are determined only by ρ.

Suppose that $A^{\mathcal{O}}(x) = f(x)$ for all $x \in \{0,1\}^n$. Then, we can obtain a deterministic inverter I for b' of the query complexity at most $t(n)$, where I simulates the oracle by using the embedded ρ, C and its own query access to b'. Particularly, I accesses b' only for answering the queries of the form $\mathcal{V}(x, \cdot)$ for some $x \in V_{n,i_{\max}(n)-1}$. However, by Lemma 9.1.11, such a bijection b' is represented by $2\log\binom{N}{a} + \log((N-a)!)$ bits, where $a = N/(t(n)+1)$, when A, ρ, and C are given. Thus, we obtain that

$$a = \frac{N}{t(n)+1} \ge \frac{2^{\Omega(n)}}{2^{O(n/\log n)}} = 2^{\Omega(n)},$$

and

$$\begin{aligned}
\Pr_{\mathcal{O}}\left[\forall x \in \{0,1\}^n, A^{\mathcal{O}}(x) = f(x) \middle| \rho, C \text{ is not bad}\right] &\le \Pr_{b'}\left[\forall x \in V_{n,i_{\max}(n)-1}, I^{b'}(x) = b'^{-1}(x) \middle| \rho, C \text{ is not bad}\right] \\
&\le \frac{\binom{N}{a}^2 \cdot (N-a)!}{N!} \\
&\le \binom{N}{a} \cdot \frac{1}{a!} \\
&\le \left(\frac{Ne}{a}\right)^a \cdot \frac{1}{\sqrt{2\pi a}} \left(\frac{e}{a}\right)^a \\
&\le (e(t(n)+1))^a \cdot \left(\frac{e}{a}\right)^a
\end{aligned}$$

$$\leq \left(\frac{2^{O(n/\log n)}}{a}\right)^a$$
$$\leq 2^{-a} = 2^{-2^{\Omega(n)}}.$$

Thus, for any choice of ρ,

$$\begin{aligned}
&\Pr_{\mathcal{O}}\left[\forall x \in \{0,1\}^n, A^{\mathcal{O}}(x) = f(x) \middle| \rho\right] \\
&\quad \leq \Pr_{\mathcal{O}}[C \text{ is bad}] + \Pr_{\mathcal{O}}\left[\forall x \in \{0,1\}^n, A^{\mathcal{O}}(x) = f(x) \middle| \rho, C \text{ is not bad}\right] \\
&\quad \leq 2^{-\Omega(t(n)^2)} + 2^{-2^{\Omega(n)}} = 2^{-\Omega(t(n)^2)}.
\end{aligned}$$

Therefore, we conclude that

$$\begin{aligned}
P_{A,n} &= \mathop{\mathbb{E}}_{\rho}\left[\Pr_{\mathcal{O}}\left[\forall x \in \{0,1\}^n,\ A^{\mathcal{O}}(x) \text{ outputs } f(x) \text{ within } t(n) \text{ steps} \middle| \rho\right]\right] \\
&\leq \mathop{\mathbb{E}}_{\rho}\left[2^{-\Omega(t(n)^2)}\right] = 2^{-\Omega(t(n)^2)}.
\end{aligned}$$

□

Corollary 9.1.13. *For any function $\epsilon(n)$ such that $\omega(1) \leq \epsilon(n) \leq n/\omega(\log^2 n)$, with probability 1 over the choice of $\mathcal{O}_{\epsilon'}$ for $\epsilon'(n) = \sqrt{\epsilon(n)}$, we have $\mathsf{UP}^{\mathcal{O}_{\epsilon'}} \cap \mathsf{coUP}^{\mathcal{O}_{\epsilon'}} \not\subseteq \mathsf{SIZE}^{\mathcal{O}_{\epsilon'}}[2^{O(\frac{n}{\epsilon(n)\log n})}]$.*

Proof. Fix a random oracle $\mathcal{O} = \mathcal{O}_{\epsilon'}$ arbitrarily. For each alphabet $y \in \Sigma_n$ ($n \in \mathbb{N}$), we use the notation $\langle y \rangle$ to refer to its unique binary expression of length $l := \lceil \log \ell(n) \rceil$. We consider the following language $L^{\mathcal{O}}$:

$$L^{\mathcal{O}} = \{(x,i) : n \in \mathbb{N}, x \in \{0,1\}^n, i \in [l], \text{ and } \exists y \in \Sigma_n \text{ s.t. } f_n(x) = y \text{ and } \langle y \rangle_i = 1\}.$$

Obviously, $y := f_n(x)$ is a unique witness for both statements $\langle x, i\rangle \in L^{\mathcal{O}}$ and $\langle x, i\rangle \notin L^{\mathcal{O}}$ by verifying whether $\mathcal{V}(x,y) = 1$ and $y_i = 1$ hold. Thus, $L^{\mathcal{O}} \in \mathsf{UP}^{\mathcal{O}} \cap \mathsf{coUP}^{\mathcal{O}}$.

Suppose that $\mathsf{UP}^{\mathcal{O}} \cap \mathsf{coUP}^{\mathcal{O}} \subseteq \mathsf{SIZE}^{\mathcal{O}}[2^{O(n/(\epsilon(n)\log n))}]$. Then, there exists a $2^{O(n/(\epsilon(n)\log n))}$-size circuit $A^{\mathcal{O}}$ that solves $L^{\mathcal{O}}$. Now, we can construct a nonuniform oracle machine $B^{\mathcal{O}}$ to compute f such as, for a given input $x \in \{0,1\}^n$, $B^{\mathcal{O}}$ executes $b_i = A^{\mathcal{O}}(\langle x, i\rangle)$ for each $i \in [l]$ and outputs $b_1 \circ \cdots \circ b_l$ by using the standard transformation from circuits to nonuniform Turing machines.

Let n' the upper bound on $|\langle x, i\rangle|$ for all $x \in \{0,1\}^n$ and $i \in [l]$, and we can assume that $n' \leq 2n$ for sufficiently large n. Then, the running time of B is bounded above by $2^{O(2n/(\epsilon(2n)\log 2n))}$, which is less than $2^{n/(\epsilon'(n)\log n)}$ for sufficiently large n. Since such an $\mathcal{O}$ contradicts the statement in Theorem 9.1.12, we conclude that the event that $\mathsf{UP}^{\mathcal{O}} \cap \mathsf{coUP}^{\mathcal{O}} \not\subseteq \mathsf{SIZE}^{\mathcal{O}}[2^{O(\frac{n}{\epsilon(n)\log n})}]$ occurs with probability 1 over the choice of $\mathcal{O}$. □

Average-Case Easiness of PH

Next, we show the average-case easiness of PH.

Theorem 9.1.14. *For any function $\omega(1) \leq \epsilon(n) \leq n/\omega(\log^2 n)$, the following event occurs with probability 1 over the choice of $\mathcal{O}_\epsilon$: for all triples of a polynomial-time oracle machine $M^?$, $d \in \mathbb{N}$, and a polynomial-time randomized oracle sampling machine $S^?$, there exists a deterministic*

polynomial-time errorless heuristic oracle machine with a failure probability of at most $n^{-\omega(1)}$ *for the distributional* Σ_d^p *problem* $(L_M^{\mathcal{O}}, \mathcal{D}_S^{\mathcal{O}})$ *determined as follows:* $(\mathcal{D}_S^{\mathcal{O}})_n \equiv S^{\mathcal{O}}(1^n)$ *for each* $n \in \mathbb{N}$ *and*

$$L_M^{\mathcal{O}} = \{x \in \{0,1\}^* : \exists w_1 \in \{0,1\}^{|x|} \forall w_2 \in \{0,1\}^{|x|}, \ldots, Q_d w_d \in \{0,1\}^{|x|}, M^{\mathcal{O}}(x, w_1, w_2, \ldots, w_d) = 1\},$$

where $Q_d = \exists$ *if* d *is an odd number; otherwise,* $Q_d = \forall$.

By the padding argument on the instance and the argument in [Imp95, Proposition 3], the above-mentioned theorem implies the following.

Corollary 9.1.15. *For any function* $\omega(1) \le \epsilon(n) \le n/\omega(\log^2 n)$, *the event* $\mathsf{DistPH}^{\mathcal{O}_\epsilon} \subseteq \mathsf{AvgP}^{\mathcal{O}_\epsilon}$ *occurs with probability* 1 *over the choice of* $\mathcal{O}_\epsilon$.

Theorem 9.1.8 immediately follows from Corollaries 9.1.13 and 9.1.15.

Proof of Theorem 9.1.14. For each n, let T_n be the maximum value of n, the square of the time for $S^?$ to generate an instance of size n, and the square of the time to execute $M^?$ on instance size n. Let $i_n = \log\log T_n$.

Now, we construct an errorless heuristic algorithm $B^{\mathcal{O}}$ that is given $x \in \{0,1\}^n$ as input and returns a value of $\mathcal{A}(\langle M, d\rangle, x, 1^{T_n^2})$. Remember that $B^{\mathcal{O}}(x) = L_M^{\mathcal{O}}(x)$ unless $\mathcal{A}(\langle M, d\rangle, x, 1^{T_n^2})$ outputs "$\perp$". Thus, we show the inequality

$$P_{n,M,S} := \Pr_{\mathcal{O},S}\left[\mathcal{A}(\langle M, d\rangle, x, 1^{T_n^2}) = \text{“}\perp\text{”} \text{ where } x \leftarrow S^{\mathcal{O}}(1^n)\right] \le n^{-\omega(1)}. \tag{9.3}$$

Then, by applying Markov's inequality, we have

$$\Pr_{\mathcal{O}}\left[\Pr_{S}\left[B^{\mathcal{O}}(x) = L_M^{\mathcal{O}}(x) \text{ where } x \leftarrow S(1^n)\right] > n^{-\omega(1)}\right] \le \frac{1}{n^2},$$

and the theorem follows from the Borel–Cantelli lemma and the countability of (M, d, S).

To show Eq. (9.3), we first show that the instance $x \in \{0,1\}^n$ is determined by only $\rho_{n',j}$ for $n' \le T_n$ and $j \le i_n - 1$ with a probability of at least $1 - n^{-\omega(1)}$. Then, we will show that $\mathcal{A}(\langle M, d\rangle, x, 1^{T_n^2})$ returns $L_M^{\mathcal{O}}(x)$ (i.e., $\mathcal{A}(\langle M, d\rangle, x, 1^{T_n^2}) \neq$“$\perp$”) with a probability of at least $1 - n^{-\omega(1)}$ under the condition that x is determined by only $\rho_{n',j}$ for $n' \le T_n$ and $j \le i_n - 1$.

Let T_n^S be the time bound for S to generate an instance of size n. Since $T_n^S \le T_n^{1/2}$, the answers of $\mathcal{A}$ to queries made by $S(1^n)$ are determined by only $\rho_{n',j}$ for $n' \le T_n^{1/2}$ and $j \le i_n - 2$. Under the condition on restrictions $\rho_{n',j}$ for $n' \le T_n^{1/2}$ and $j \le i_n - 2$, the value of $x \leftarrow S^{\mathcal{O}}(1^n)$ is determined by only $\rho_{n',j}$ for $n' \le T_n$ and $j \le i_n - 1$ unless S queries $x \in \{0,1\}^{\le T_n^{1/2}}$ such that $F_x \in V_{|x|,i_n-1}$ to $\mathcal{V}$. Note that, if $n' \in \mathbb{N}$ satisfies $n' < t^{-1}(T_n^{1/2})$, then we have $i_{\max}(n') < \log\log(T_n^{1/2}) = \log\log T_n - 1 = i_n - 1$. Thus, $V_{n',i_n-1} = \emptyset$. Otherwise, $V_{|x|,i_n-1}$ is selected from $V_{|x|,i_n-2}$ uniformly at random. Thus, such a conditional probability is bounded above by

$$\begin{aligned} T_n^{1/2} \max_{t^{-1}(T_n^{1/2}) \le n' \le T_n^{1/2}} \frac{\lfloor p(n')|V_{n',i_n-2}|\rfloor}{|V_{n',i_n-2}|} &= O\left(T_n^{1/2} p(t^{-1}(T_n^{1/2}))\right) \\ &= O\left(T_n^{1/2} \cdot (T_n^{1/2})^{-\sqrt{\epsilon(n)}}\right) \\ &= T_n^{-\omega(1)} \\ &= n^{-\omega(1)}, \end{aligned}$$

where the last equation holds because $T_n \geq n$.

Under the condition that the given instance $x \in \{0,1\}^n$ is determined by only $\rho_{n',j}$ for $n' \leq T_n$ and $j \leq i_n - 1$, the depth $d+2$ circuit C constructed during the execution of $\mathcal{A}(\langle M, d\rangle, x, 1^{T_n^2})$ is determined only by $\rho_{n',j}$ for $n' \leq T_n$ and $j \leq i_n - 1$. Then, applying the restriction $\rho_{n',j}$ for $n' \leq T_n$ and $j \leq i_n$ under this condition is regarded as a $p(n')$-random restriction for V_{n',i_n-1} for each $n' \leq T_n$, where we can ignore small n' such that $n' < t^{-1}(T_n)$ because $i_{\max}(n') < \log\log T_n = i_n$ for such n'. Note that the width and internal size of C are at most T_n and 2^{T_n+1}, respectively. Thus, by applying Lemma 9.1.7, the probability that C does not become a constant (i.e., the probability that $\mathcal{A}$ returns "$\perp$") is at most

$$\begin{aligned} O\left(T_n^2 \max_{t^{-1}(T_n)\leq n'\leq T_n} p(n')^{1/(d+2)}\ell(n')^2\right) &= O\left(T_n^2 \max_{t^{-1}(T_n)\leq n'\leq T_n} t(n')^{-\frac{\omega(1)}{d+2}+4}\right) \\ &= T_n^{-\omega(1)} \\ &= n^{-\omega(1)}, \end{aligned}$$

where the last equation holds because $T_n \geq n$. □

Oracle Separation between UP ∩ coUP and Distributional Σ_d^p

Theorem 9.1.16. *For any constant $a > 0$ and $d \in \mathbb{N}$, there exists an oracle $\mathcal{O}_{a,d}$ satisfying (1) $\mathsf{Dist}\Sigma_d^{p\mathcal{O}_{a,d}} \subseteq \mathsf{AvgP}^{\mathcal{O}_{a,d}}$ and (2) $\mathsf{UP}^{\mathcal{O}_{a,d}} \cap \mathsf{coUP}^{\mathcal{O}_{a,d}} \not\subseteq \mathsf{SIZE}^{\mathcal{O}_{a,d}}[2^{\frac{an}{\log n}}]$.*

Proof sketch. Let $c = \max\{21(d+2)a, 1\}$ and $\epsilon(n) = 1/a$ for each $n \in \mathbb{N}$. We construct an oracle $\mathcal{O}_{a,d}$, as in Section 9.1.3, where we set the parameters as follows:

$$t(n) = 2^{\frac{n}{\epsilon(n)\cdot\log n}},\ p(n) = t(n)^{-5(d+2)},\ \ell(n) = t(n)^2,\ \text{and}\ i_{\max}(n) = \frac{1}{c}\log\log t(n).$$

We also change the simulation overhead from T^2 to T^{2^c} and the setting of i from $\log\log n$ to $c^{-1}\log\log n$ in $\mathcal{A}$. Then, we can easily show the analog of Lemma 9.1.9. Further, we get

$$p(n)^{i_{\max}(n)-1} \geq t(n)^{-\frac{5(d+2)\log n}{c}} = 2^{-\frac{20(d+2)an\log n}{c\log n}} \geq 2^{-\frac{20}{21}n},$$

and $|V_{n,i_{\max}(n)}| = \Omega(p(n)^{i_{\max}(n)-1}\cdot 2^n) = 2^{\Omega(n)}$. Thus, we can show the hardness of computing in $t(n) = 2^{\frac{n}{\epsilon(n)\cdot\log n}} = 2^{an/\log n}$ steps using the same proof as that of Theorem 9.1.12. It is not hard to verify that this lower bound yields $\mathsf{UP}^{\mathcal{O}_{a',d}} \cap \mathsf{coUP}^{\mathcal{O}_{a',d}} \not\subseteq \mathsf{BPTIME}^{\mathcal{O}_{a',d}}[2^{\frac{an}{\log n}}]$ by a proper choice of $a' > 0$ in the same way as Corollary 9.1.13. The average-case easiness of $\mathsf{Dist}\Sigma_d^p$ also holds by the same argument to as proof of Theorem 9.1.14, where we select T_n as the maximum value of n^4, n^{2^c}, the 2^c-th power of the time for $S^?$ to generate an instance of size n, and the 2^c-th power of the time to execute $M^?$ on instance size n (see also Section 9.1.4). □

9.1.4 Oracle Separation: Learning and Distributional PH

We prove the oracle separation between the hardness of learning and distributional PH. In this section, we present Theorem 9.1.17. Note that we can obtain the second item of Theorem 9.1.1 as a corollary to Theorem 9.1.17 (i.e., Corollary 9.1.18). The first item of Theorem 9.1.1 is shown in a similar way by changing the parameters (we will discuss this after proving Theorem 9.1.17).

Theorem 9.1.17. *Let $a : \mathbb{N} \to \mathbb{R}_{>0}$ be a function such that $n/\omega(\log^2 n) \leq a(n) \leq O(1)$. For any $d \in \mathbb{N}$ and sufficiently large $c \in \mathbb{N}$, there exists an oracle $\mathcal{O} := \mathcal{O}_{a,c,d}$ such that (1) $\mathsf{Dist}\Sigma_d^{p\mathcal{O}} \subseteq \mathsf{AvgP}^{\mathcal{O}}$ and (2) no nonuniform $2^{\frac{a(n)n}{\log n}}$-time learner can weakly PAC learn $\mathsf{SIZE}^{\mathcal{O}}[n]$ with membership queries on $\mathscr{D}$, where $\mathscr{D}$ is an arbitrary class of example distributions such that $\mathscr{D}_n$ contains all uniform distributions over subsets $S \subseteq \{0,1\}^n$ with $|S| \geq 2^{(1-\frac{a(n)}{c})\cdot n}$.*

The following immediately follows from Theorem 9.1.17.

Corollary 9.1.18. *For any constants $a, \epsilon > 0$ and $d \in \mathbb{N}$, there exists an oracle $\mathcal{O}$ such that (1) $\mathsf{Dist}\Sigma_d^{p\mathcal{O}} \subseteq \mathsf{AvgP}^{\mathcal{O}}$ and (2) no nonuniform $2^{\frac{an}{\log n}}$-time learner can weakly PAC learn $\mathsf{SIZE}^{\mathcal{O}}[n]$ with membership queries on all uniform distributions over subsets $S \subseteq \{0,1\}^n$ with $|S| \geq 2^{(1-\epsilon)\cdot n}$.*

Construction of Random Oracle

Let $\epsilon\colon \mathbb{N} \to \mathbb{N}$ and $c, d \in \mathbb{N}$ denote parameters satisfying that $\Omega(1) \leq \epsilon(n) \leq n/\omega(\log^2 n)$ and $c \geq \max\{3, 26(d+2)/\epsilon(n)\}$ for sufficiently large n.

Construction. *$\mathcal{O}_{\epsilon,c,d} = \mathcal{F} + \mathcal{A}$, where each oracle is randomly selected by the following procedure:*

1. *Define functions t, p, ℓ, q, and $i_{\max}$ as*

$$t(n) = 2^{\frac{n}{\epsilon(n)\cdot\log n}},\ p(n) = t(n)^{-11(d+2)},\ \ell(n) = t(n)^4, q(n) = t(n)^{-3(d+2)},\ \text{and } i_{\max}(n) = \frac{1}{c}\log\log t(n).$$

2. *For each $n \in \mathbb{N}$, define a set $V_{n,0}$ of variables on alphabet Σ_n of size $\ell(n)$ as*

$$V_{n,0} = \{F_z : z \in \{0,1\}^n\}.$$

   We assume that each alphabet in Σ_n has a binary representation of length at most $\lceil \log \ell(n) \rceil$.

3. *For each $n \in \mathbb{N}$, define a set $W_{n,0}$ of variables on alphabet $\{0,1\}$ as*

$$W_{n,0} = \{G_{z,x} : z, x \in \{0,1\}^n\}.$$

4. *For each $n \in \mathbb{N}$ and $i \in [i_{\max}(n) - 1]$, we inductively (on i) define a $p(n)$-random restriction $\rho_{n,i}$ to $V_{n,i-1}$, a $q(n)$-random restriction $\sigma_{n,i}$ to $W_{n,i-1}$, and subsets $V_{n,i} \subset V_{n,i-1}$ and $W_{n,i} \subset W_{n,i-1}$ of variables as*

$$V_{n,i} = \{v \in V_{n,i-1} : \rho_{n,i}(v) = *\} \text{ and } W_{n,i} = \{w \in W_{n,i-1} : \sigma_{n,i}(w) = *\}.$$

   We also define $\rho_{n,i_{\max}(n)}$ (resp. $\sigma_{n,i_{\max}(n)}$) as a 0-random restriction (i.e., a full assignment) to $V_{n,i_{\max}(n)-1}$ (resp. $W_{n,i_{\max}(n)-1}$). Let $\rho_{n,i} \equiv \rho_{n,i_{\max}(n)}$ and $\sigma_{n,i} \equiv \sigma_{n,i_{\max}(n)}$ for $i \geq i_{\max}(n)+1$. For simplicity, we may identify $\rho_{n,i}$ (resp. $\sigma_{n,i}$) with the composite restriction $\rho_{n,1}\cdots\rho_{n,i}$ to $V_{n,0}$ (resp. $\sigma_{n,1}\cdots\sigma_{n,i}$ to $W_{n,0}$) for each n and i.

5. *Let $f = \{f_n\}_{n\in\mathbb{N}}$, where $f_n\colon \{0,1\}^n \to \Sigma_n$ is a random function defined as $f_n(z) = \rho_{n,i_{\max}(n)}(F_z)$. Let $g = \{g_n\}_{n\in\mathbb{N}}$, where $g_n\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ is a random function defined as $g_n(z,x) = \sigma_{n,i_{\max}(n)}(G_{z,x})$.*

6. *Define* $\mathcal{F} = \{\mathcal{F}_n\}_{n\in\mathbb{N}}$, *where* $\mathcal{F}_n\colon \{0,1\}^n \times \Sigma_n \times \{0,1\}^n$, *as follows:*

$$\mathcal{F}_n(z,y,x) = \begin{cases} g_n(z,x) & \text{if } y = f_n(z) \\ 0 & \text{otherwise.} \end{cases}$$

7. *Define* $\mathcal{A}$ *as follows: On input* $(\langle M, d\rangle, x, 1^{T^{2^c}})$, *where* M *is an oracle machine,* $d \in \mathbb{N}$, $x \in \{0,1\}^*$, *and* $T \in \mathbb{N}$, *the oracle* $\mathcal{A}$ *returns the value in* $\{0,1,\perp\}$ *determined according to the following procedure:*

   1: Let $i := \frac{1}{c}\log\log T$.
   2: Construct a depth $d+2$ *circuit* C *corresponding to the quantified formula*

   $$\exists w_1 \in \{0,1\}^{|x|} \forall w_2 \in \{0,1\}^{|x|}, \ldots, Q_d w_d \in \{0,1\}^{|x|}, M^{\mathcal{O}}(x, w_1, w_2, \ldots, w_d),$$

   where $Q_d = \exists$ *if* d *is an odd number; otherwise,* $Q_d = \forall$.
   First, we construct a depth-d circuit that represents the above-mentioned quantified formula whose leaf corresponds to $M^{\mathcal{O}}(x, w_1, w_2, \ldots, w_d)$ *for some* $w_1, w_2, \ldots, w_d$, *where we truncate* $w_1, w_2, \ldots, w_d$ *into a string of length* T *because we will execute* M *in only* T *steps. Then, we replace each leaf with a DNF formula of width* $2T$ *to obtain the circuit* C, *where each term corresponds to one possible choice of* $\mathcal{F}$ *such that* $M^{\mathcal{F}+\mathcal{A}}(x, w_1, w_2, \ldots, w_d)$ *halts with an accepting state after execution in* T *steps. In other words, we arbitrarily consider functions* $f' = \{f'_n\}_{n=1}^T$ *and* $g' = \{g'_n\}_{n=1}^T$, *where* $f'_n\colon \{0,1\}^n \to \Sigma_n$ *and* $g'_n\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$, *define an oracle* $\mathcal{F}'$ *in the same manner as* $\mathcal{F}$, *and execute* $M^{\mathcal{F}'+\mathcal{A}}(x, w_1, w_2, \ldots, w_d)$ *in* T *steps. If* M *queries* (z,y,x) *to* $\mathcal{F}'$, *then we add literals to the corresponding term in the following manner:*

   $$\textit{add literals} \begin{cases} (F_z \neq y) & \text{if } f'(z) \neq y \\ (F_z = y) \text{ and } (G_{z,x} = 0) & \text{if } f'(z) = y \text{ and } g'(z,x) = 0 \\ (F_z = y) \text{ and } (G_{z,x} = 1) & \text{if } f'(z) = y \text{ and } g'(z,x) = 1 \end{cases}$$

   By the construction, the above-mentioned quantified formula is satisfied when $M^{\mathcal{F}+\mathcal{A}}$ *is executed in* T *steps iff* C *returns* 1 *when it is restricted by* $\rho_{1,i_{\max}(1)}, \ldots, \rho_{T,i_{\max}(T)}$ *and* $\sigma_{1,i_{\max}(1)}, \ldots, \sigma_{T,i_{\max}(T)}$. *We can also easily verify that the width of* C *is at most* $2T$, *and the internal size of* C *is at most* 2^{T+1}.
   3: If $C|_{\rho_{1,i},\ldots,\rho_{T,i}} \equiv b$ *for some* $b \in \{0,1\}$, *then* ***return*** b; *otherwise,* ***return*** *"*$\perp$*".*

To verify that $\mathcal{A}$ is not circular on recursive calls for $\mathcal{A}$, it is sufficient to check the following:

Lemma 9.1.19. *For each input, the value of* $\mathcal{A}(\langle M, d\rangle, x, 1^{T^{2^c}})$ *is determined only by* $\rho_{n,j}$ *and* $\sigma_{n,j}$ *for* $n \leq T$ *and* $j \leq \frac{1}{c}\log\log T\,(= i)$.

Proof. We show the lemma by induction on T. We consider the execution of $(\langle M, d\rangle, x, 1^{T^{2^c}})$. We remark that $\mathcal{A}$ first makes a depth $d+2$ circuit C based on M, and C is independent of the value of $\mathcal{F}$.

We assume that M makes some valid query $(\langle M', d'\rangle, x', 1^{T'^{2^c}})$ to $\mathcal{A}$ recursively on constructing C. Since the length of such a query is at most T, we have $T'^{2^c} \leq T$. If we let $i' = c^{-1} \log\log T'$, then we have

$$i' = \frac{1}{c}\log\log T' \leq \frac{1}{c}\log\log T^{\frac{1}{2^c}} = \frac{1}{c}\log\log T - 1.$$

By the induction hypothesis, the recursive answer of $\mathcal{A}$ is determined by only $\rho_{n,j}$ and $\sigma_{n,j}$ for $n \leq T'$ and $j \leq i-1$, and so is C. The lemma holds because the answer of $\mathcal{A}$ is determined by restricting C by $\rho_{n,j}$ and $\sigma_{n,j}$ for $n \leq T$ and $j \leq i$. □

Note that we may omit the subscripts ϵ, c, and d from $\mathcal{O}_{\epsilon,c,d}$.

Hardness of Learning

First, we show the hardness of learning.

Theorem 9.1.20. *For arbitrary parameters $\epsilon(n)$, c, and d such that $\Omega(1) \leq \epsilon(n) \leq n/\omega(\log^2 n)$ and $c \geq \max\{3, 26(d+2)/\epsilon(n)\}$ (for sufficiently large n), the following event occurs with probability 1 over the choices of $\mathcal{O} := \mathcal{O}_{\epsilon,c,d}$: a concept class $\mathscr{C}^{\mathcal{O}}$ defined as*

$$\mathscr{C}^{\mathcal{O}} = \{\mathcal{F}_n(z, y, \cdot) : n \in \mathbb{N}, z \in \{0,1\}^n, y \in \Sigma_n\}$$

is not weakly PAC learnable (with confidence error at most $1/3$) on $\mathscr{D}$ by nonuniform $t(n) = 2^{\frac{n}{\epsilon(n)\log n}}$-time algorithms, where $\mathscr{D} = \{\mathscr{D}_n\}_{n\in\mathbb{N}}$ is a class of example distributions such that each $\mathscr{D}_n$ contains all uniform distributions over subsets $S \subseteq \{0,1\}^n$ with $|S| \geq 2^{(1-\frac{4(d+2)}{c\epsilon(n)})\cdot n}$.

To show Theorem 9.1.20, we use the following lemma.

Lemma 9.1.21. *Let U be a universe of size N, and let $Z \subseteq U$ be an arbitrary subset of size M ($\leq N$). Let $S \subseteq U$ be a random subset of size n. Then, for any $\gamma \in (0,1)$, we have*

$$\Pr_T\left[\left||S \cap Z| - \frac{M}{N}n\right| > \gamma \cdot \frac{M}{N}n\right] < 2e^{-2\gamma^2\cdot(\frac{M}{N})^2\cdot n}.$$

We defer the proof of Lemma 9.1.21 to the end of this section. Now, we present the formal proof of Theorem 9.1.20.

Proof of Theorem 9.1.20. For any choice of $\mathcal{O}$ and $z \in \{0,1\}^n$, we define a subset $G_z^* \subseteq \{0,1\}^n$ as

$$G_z^* = \{x \in \{0,1\}^n : \rho_{n,i_{\max}(n)-1}(G_{z,x}) = *\}.$$

We let U_z^* denote a uniform distribution over the elements in G_z^*. For $z \in \{0,1\}^n$, we define a function $g_z : \{0,1\}^n \to \{0,1\}$ as $g_z(x) = g(z,x)$ $(= \mathcal{F}_n(z, f_n(z), x) \in \mathscr{C}^{\mathcal{O}})$.

We say that $z \in \{0,1\}^n$ is a "hard" index if $z \in V_{n,i_{\max}(n)-1}$ and $|G_z^*| \geq 2^{(1-\frac{4(d+2)}{c\epsilon(n)})\cdot n}$. We show that f_z is hard to learn on example distribution U_z^* for a hard index z.

First, we estimate the probability that such a hard index exists.

Claim 9.1.22. *If $\epsilon(n) \geq \Omega(1)$ and $c \geq 26(d+2)/\epsilon(n)$, then we have*

$$\Pr_{\mathcal{O}}[\textit{there exists no hard index in } \{0,1\}^n] \leq 2^{-2^{\Omega(n)}}.$$

Proof. Since $\epsilon(n) \geq \Omega(1)$, we have $t(n) \leq 2^n$ and $i_{\max}(n) \leq \frac{1}{c}\log n$ for sufficiently large n. Thus, we have that for sufficiently large n,

$$p(n)^{i_{\max}(n)} \geq t(n)^{-\frac{11(d+2)}{c}\log n} = 2^{-\frac{11(d+2)n\log n}{c\epsilon(n)\log n}} \geq 2^{-\frac{11}{26}n},$$

and

$$q(n)^{i_{\max}(n)} \geq t(n)^{-\frac{3}{c}\log n} = 2^{-\frac{3n\log n}{c\epsilon(n)\log n}} \geq 2^{-\frac{3}{26}n}.$$

By Lemma 9.1.21, for any $z \in \{0,1\}^n$, we get

$$\begin{aligned}\Pr\left[\sum_{z\in V_{n,i_{\max}(n)-1}} |G_z^*| < \frac{1}{2}\cdot\frac{|V_{n,i_{\max}(n)-1}|\cdot 2^n}{2^{2n}}|W_{n,i_{\max}(n)-1}|\right] &< 2\exp\left(-\Omega(p(n)^{2i_{\max}(n)}q(n)^{i_{\max}(n)})\cdot 2^{2n}\right)\\ &\leq 2\exp\left(-\Omega(2^{-\frac{22}{26}n}\cdot 2^{-\frac{3}{26}n}\cdot 2^{2n})\right)\\ &\leq 2^{-2^{\Omega(n)}}.\end{aligned}$$

If the above-mentioned event does not occur, then there exists $z \in V_{n,i_{\max}(n)-1}$ such that

$$|G_z^*| \geq \frac{|W_{n,i_{\max}(n)-1}|}{2^{n+1}} = \Omega(q(n)^{i_{\max}(n)}\cdot 2^n) = \Omega(2^{n-\frac{3(d+2)n}{c\epsilon(n)}}).$$

Thus, $|G_z^*| \geq 2^{n-\frac{4(d+2)n}{c\epsilon(n)}}$ for sufficiently large $n \in \mathbb{N}$, and such z is a hard index. ◇

For Theorem 9.1.20, we show that for every nonuniform $t(n)$-time learning algorithm L and every sufficiently large $n \in \mathbb{N}$,

$$\Pr_{\mathcal{O}}\left[\forall z \in \{0,1\}^n \text{ and } \forall \mathcal{D} \in \mathscr{D}_n, \Pr_{L,S}\left[L^{\mathcal{O},g_z}(S) \to h^{\mathcal{O}} \text{ s.t. } \Pr_{x\sim\mathcal{D}}[h^{\mathcal{O}}(x) = g_z(x)] \geq \frac{1}{2}+\delta_n\right] \geq \frac{2}{3}\right] \leq 2^{-2^{\Omega(n)}}, \tag{9.4}$$

where $\delta_n = t(n)^{-1/4}$ ($\geq n^{-\omega(1)}$), and S is a sample set of size at most $t(n)$ generated according to $\mathsf{EX}_{g_z,\mathcal{D}}$.

If Eq. (9.4) holds, then we can derive Theorem 9.1.20 as follows. Since any nonuniform $t(n)$-time algorithm can be described by $O(t(n))$-bits, by the union bound, the event in Eq. (9.4) does not hold for all nonuniform $t(n)$-time algorithms and all sufficiently large $n \in \mathbb{N}$ with probability at least $1 - 2^{-2^{\Omega(n)}}\cdot 2^{O(t(n))} = 1 - 2^{-2^{\Omega(n)}}$ over the choice of $\mathcal{O}$. By the Borel–Cantelli lemma, with probability 1 over the choice of $\mathcal{O}$, it holds that for all sufficiently large $n \in \mathbb{N}$, there is no nonuniform $t(n)$-time algorithm L satisfying that

$$\forall z \in \{0,1\}^n \text{ and } \forall \mathcal{D} \in \mathscr{D}_n, \Pr_{L,S}\left[L^{\mathcal{O},g_z}(S) \to h^{\mathcal{O}} \text{ s.t. } \Pr_{x\sim\mathcal{D}}[h^{\mathcal{O}}(x) = g_z(x)] \geq \frac{1}{2}+\delta_n\right] \geq \frac{2}{3}.$$

Thus, it suffices to show Eq. (9.4).

For $z \in \{0,1\}^n$, $x \in \{0,1\}^n$, and a sample set S, we use the notation $L^{\mathcal{O},g_z}(S)(x)$ to refer to the following procedure: (1) execute $L^{\mathcal{O},g_z}(S)$; (2) if L outputs some hypothesis h within $t(n)$ steps,

then execute $h^{\mathcal{O}}(x)$. For $z \in \{0,1\}^n$, we define events I_z and J_z (over the choice of $\mathcal{O}$) as follows:

$$I_z = \left(\Pr_{L,S,x} \left[\mathcal{F}(z, f_n(z), x') \text{ is queried for some } x' \in \{0,1\}^n \text{ during } L^{\mathcal{O},g_z}(S)(x) \right] \geq \delta_n^4 \right)$$

$$J_z = \left(\Pr_{L,S} \left[L^{\mathcal{O},g_z}(S) \to h^{\mathcal{O}} \text{ s.t. } \Pr_x[h^{\mathcal{O}}(x) = g_z(x)] \geq \frac{1}{2} + \delta_n \text{ within } t(n) \text{ steps} \right] \geq \frac{2}{3} \right),$$

where S is selected according to EX_{g_z,U_z^*}, and x is selected according to U_z^*.

We assume that $z \in \{0,1\}^n$ is a hard index. Then, we have $|G_z^*| \geq 2^{(1-\frac{4(d+2)}{c\epsilon(n)})\cdot n}$; thus, U_z^* must be contained in $\mathscr{D}_n$. Therefore, the left-hand side of Eq. (9.4) is bounded above by

$$\begin{aligned}
\Pr_{\mathcal{O}}\left[\bigwedge_{z:\text{hard}} J_z\right] &\leq \Pr_{\mathcal{O}}\left[\bigwedge_{z:\text{hard}} J_z \vee I_z\right] \\
&\leq \Pr_{\mathcal{O}}\left[\left(\bigwedge_{z:\text{hard}} I_z\right) \vee \left(\bigvee_{z:\text{hard}} J_z \wedge \neg I_z\right)\right] \\
&\leq \Pr_{\mathcal{O}}\left[\bigwedge_{z:\text{hard}} I_z\right] + \Pr_{\mathcal{O}}\left[\bigvee_{z:\text{hard}} J_z \wedge \neg I_z\right]. \qquad (9.5)
\end{aligned}$$

Here, we let P_1 and P_2 represent the first and second terms of Eq. (9.5), respectively. We derive the upper bounds on P_1 and P_2 as the following claims, which immediately imply Eq. (9.4).

Claim 9.1.23. $P_1 = \Pr_{\mathcal{O}}\left[\bigwedge_{z:hard} I_z\right] \leq 2^{-2^{\Omega(n)}}$.

Claim 9.1.24. $P_2 = \Pr_{\mathcal{O}}\left[\bigvee_{z:hard} J_z \wedge \neg I_z\right] \leq 2^{-2^{\Omega(n)}}$.

Proof of Claim 9.1.23. First, we fix random restrictions except for $\rho_{n,i_{\max}(n)}$ and use π to denote the composite restriction. Note that all the hard indices are determined at this stage. Assume that there exists a hard index of length n, and let $z_\pi \in \{0,1\}^n$ be the lexicographically first hard index. Then, we can divide $\rho_{n,i_{\max}(n)}$ into two random selections as follows. First, we randomly select unassigned values of $f_n(z)$ except for $f_n(z_\pi)$ (let π' denote the corresponding random restriction). Then, we select the remaining value of $f_n(z_\pi)$ from Σ_n uniformly at random.

We remark that π determines g and G_z^* for all $z \in \{0,1\}^n$. Now, we construct a randomized oracle machine A to compute $f_n(z_\pi)$ based on L, g, G_z^*, π, π', and additional oracle access to $\mathcal{V}$, where $\mathcal{V}(y)$ returns 1 if $y = f_n(z_\pi)$ (otherwise, returns 0).

On input z_π and oracle access to $\mathcal{V}$, A executes L in $t(n)$ steps for a target function g_z and an example distribution U_z^* (note that examples and membership queries are simulated by g and G_z^*); if L outputs some hypothesis h, then compute $h(x)$ for $x \sim U_z^*$, where A answers the queries of L and h to $\mathcal{O}$ as follows:

> $\mathcal{F}(z,y,x)$: If $z = z_\pi$, then A queries y to $\mathcal{V}$; if $\mathcal{V}$ returns 1, then return $g_n(z,x)$ (otherwise, return 0). In other cases, A can correctly answer $\mathcal{F}(z,y,x)$ because it is determined by π and π'.
> $\mathcal{A}(\langle M,d\rangle, x, 1^{T^{2^c}})$: Since A executes L only $t(n)$ steps, we can assume that the size of h is at most $t(n)$ and it is evaluated in time $O(t(n)^2)$. Thus, we can assume that $T^{2^c} = O(t(n)^2)$ and for sufficiently large n,
>
> $$i = \frac{1}{c}\log\log T = \frac{1}{c}\log\log O(t(n))^{1/2^{c-1}} \leq \frac{1}{c}\log\log t(n)^{1/2^{c-2}} \leq \frac{1}{c}\log\log t(n) - \frac{c-2}{c},$$

which is strictly smaller than $i_{\max}(n) (= \frac{1}{c} \log \log t(n))$ because $c \geq 3$. By Lemma 9.1.19, the answer does not depend on $\rho_{n,i_{\max}(n)}$. Thus, A can correctly simulate $\mathcal{A}$ by π and π' in this case.

A repeats the above-mentioned executions of L and its hypothesis n/δ_n^4 times. If A queries y such that $\mathcal{V}(y) = 1$ at some trial, then A outputs y $(= f_n(z_\pi))$ and halts (otherwise, A outputs $\perp$).

By the construction, A can correctly simulate L and its hypothesis h for a target function g_z and an example distribution U_z^*. It is easy to verify that the number q of the queries of A to $\mathcal{V}$ is bounded as $q \leq (n/\delta_n^4) \cdot O(t(n)^2) \leq O(n) \cdot t(n)^3$.

Assume that $\wedge_{z:\text{hard}} I_z$ holds. Then, L or h queries $(z_\pi, f_n(z_\pi), \cdot)$ to $\mathcal{F}$ with a probability of at least δ_n^4 for each trial. Since A repeats this trial n/δ_n^4-times, the failure probability of A is at most $(1-\delta_n^4)^{n/\delta_n^4} < 2^{-n}$. Thus, we have

$$\Pr_{\mathcal{O},A}\left[A^{\mathcal{V}}(z_\pi) = f_n(z_\pi) \middle| \pi, \pi', \wedge_{z:\text{hard}} I_z\right] \geq 1 - 2^n. \tag{9.6}$$

Meanwhile, even under the condition on π and π', the value of $f_n(z_\pi)$ is selected from Σ_n at random independently of A. Thus, we can also show that

$$\Pr_{\mathcal{O},A}\left[A^{\mathcal{V}}(z_\pi) = f_n(z_\pi) \middle| \pi, \pi', \wedge_{z:\text{hard}} I_z\right] \leq \frac{q}{\ell(n)} = \frac{O(n)}{t(n)} = n^{-\omega(1)}. \tag{9.7}$$

Eq. (9.7) contradicts Eq. (9.6). This indicates that there exists no hard index in this case. By Claim 9.1.22, we conclude that

$$P_1 = \Pr_{\mathcal{O}}\left[\bigwedge_{z:\text{hard}} I_z\right] \leq \Pr_{\mathcal{O}}\left[\text{there exists no hard index in } \{0,1\}^n\right] \leq 2^{-2^{\Omega(n)}}.$$

◇

Proof of Claim 9.1.24. We fix $z \in \{0,1\}^n$ arbitrarily, and we let $\mathcal{O}'$ denote a partial choice of $\mathcal{O}$ except for the values of $g(z,x)$, where $x \in G_z^*$. Then, we have

$$\Pr_{\mathcal{O}}\left[\neg I_z \wedge J_z\right] = \mathop{\mathbb{E}}_{\mathcal{O}'}\left[\Pr_{\mathcal{O}}\left[\neg I_z \wedge J_z | \mathcal{O}'\right]\right].$$

We remark that g_z is a truly random (partial) function on G_z^*, even under the condition on $\mathcal{O}'$.

Assume that z is a hard index, and $\neg I_z \wedge J_z$ occurs. Let $N = |G_z^*|$. Since z is a hard index, $N \geq 2^{(1-\frac{4(d+2)}{c\epsilon(n)})\cdot n} \geq 2^{\Omega(n)}$ holds.

By Markov's inequality and $\neg I_z$, we have

$$\Pr_{L,S}\left[\Pr_x\left[\mathcal{F} \text{ is asked } (z, f_n(z), \cdot) \text{ during } L^{\mathcal{O},g_z}(S)(x)\right] \leq 4\delta_n^3\right] \geq 1 - \frac{\delta_n}{4}.$$

By J_z, we also get

$$\Pr_{L,S}\left[L^{\mathcal{O},g_z}(S) \to h^{\mathcal{O}} \text{ s.t. } \Pr_x[h^{\mathcal{O}}(x) = g_z(x)] \geq \frac{1}{2} + \delta_n \text{ within } t(n) \text{ steps}\right] \geq \frac{2}{3}.$$

By the two above-mentioned inequalities, there exist a sample set S and a random string r for L such that

- $L^{\mathcal{O},g_z}(S;r)$ outputs some hypothesis $h^{\mathcal{O}}$ in time $t(n)$ without querying $(z, f_n(z), \cdot)$ to $\mathcal{F}$;
- $\Pr_{x\sim U_z^*}[h^{\mathcal{O}}(x)$ queries $(z, f_n(z), \cdot)$ to $\mathcal{F}] \le 4\delta_n^3$; and
- $\Pr_{x\sim U_z^*}[h^{\mathcal{O}}(x) = f_z(x)] \ge \frac{1}{2} + \delta_n$.

If L and h do not query $(z, f_n(z), \cdot)$ to $\mathcal{F}$ and they halt in $t(n)$ steps, then the answers by $\mathcal{O}$ do not depend on $\sigma_{n,i_{\max}(n)}$, i.e., the values of $g_z(x)$ for $x \in G_z^*$, as seen in the proof of Claim 9.1.23. In other words, they are totally determined by $\mathcal{O}'$. Thus, we can replace $\mathcal{O}$ with $\mathcal{O}'$ in these cases (where we assume that $\mathcal{O}'$ returns an error on an undefined input).

Now, we show that a truth table $\tau \in \{0,1\}^N$ of g_z on G_z^* has a short description (under the condition on $\mathcal{O}'$), which yields the upper bound on P_2 because a random function does not have such a short description with high probability.

Let $B_z \subseteq G_z^*$ be the subset consisting of x such that $h^{\mathcal{O}}(x)$ queries $(z, f_n(z), \cdot)$ to $\mathcal{F}$. By the second property, we have $|B_z| \le 4N\delta_n^3$. We consider the following reconstruction procedure for τ. First, we execute $L^{\mathcal{O}',g_z}(S;r)$ to obtain $h^{\mathcal{O}'}$. Note that if we obtain all answers for membership queries by L as auxiliary advice Q (of length at most $t(n)$), then we can remove external access to g_z from L. Next, we execute $h^{\mathcal{O}'}(x)$ on each input $x \in G_z^* \setminus B_z$. By combining these predictions with auxiliary advice $S_B = \{(x, g_z(x)) : x \in B_z\}$, we also obtain a partial truth table $\tilde{\tau} \in \{0,1\}^N$ $(1/2 - \delta_n)$-close to $\tau \in \{0,1\}^N$. If we obtain $err \in \{0,1\}^N$ defined as $err_i = \tau_i \oplus \tilde{\tau}_i$ as auxiliary advice, then we can reconstruct τ from $\tilde{\tau}$ and err.

Therefore, we can reconstruct τ from L, S, r, Q, S_B, and err under the condition on $\mathcal{O}'$. Since the Hamming weight of err is at most $N \cdot (1/2 + \delta_n)$, err is represented by a binary string of length at most $(1 - \Omega(\delta_n^2)) \cdot N$ by lexicographic indexing among binary strings of the same weight. Hence, τ has a short description of length at most

$$O(t(n)) + 4\delta_n^3(n+1) \cdot N + (1 - \Omega(\delta_n^2)) \cdot N \le O(t(n)) + \left(1 - \Omega(\delta_n^2)\right) \cdot N.$$

Since τ is a truly random string even under the condition on $\mathcal{O}'$, we have

$$\begin{aligned}
\Pr_{\mathcal{O}}\left[z \text{ is hard and } \neg I_z \wedge J_z | \mathcal{O}'\right] &\le \frac{2^{O(t(n)) + \left(1-\Omega(\delta_n^2)\right)\cdot N}}{2^N} \\
&\le 2^{O(t(n)) - \Omega(t(n)^{-1/2})\cdot N} \\
&\le 2^{2^{O(n/\log n)} - 2^{\Omega(n - n/\log n)}} \\
&\le 2^{-2^{\Omega(n)}}.
\end{aligned}$$

This implies that $\Pr_{\mathcal{O}}[z \text{ is hard and } \neg I_z \wedge J_z] \le \mathbb{E}_{\mathcal{O}'}[\Pr_{\mathcal{O}}[z \text{ is hard and } \neg I_z \wedge J_z | \mathcal{O}']] \le 2^{-2^{\Omega(n)}}$ for any index z. Note that the number of indices is at most 2^n. Thus, by taking the union bound, we conclude that

$$P_2 = \Pr_{\mathcal{O}}\left[\bigvee_{z:\text{hard}} J_z \wedge \neg I_z\right] \le 2^n \cdot 2^{-2^{\Omega(n)}} = 2^{-2^{\Omega(n)}}.$$

$\diamond$

$\square$

Average-Case Easiness of Σ_d^p

Next, we show the average-case easiness of Σ_d^p.

Theorem 9.1.25. *For any parameters $\epsilon(n)$, c, and d such that $\Omega(1) \le \epsilon(n) \le n/\omega(\log^2 n)$ and $c \ge \max\{3, 26(d+2)/\epsilon(n)\}$ (for sufficiently large n), the following event occurs with probability 1 over the choice of $\mathcal{O} := \mathcal{O}_{\epsilon,c,d}$: for all tuples of a polynomial-time oracle machine $M^?$ and a polynomial-time randomized oracle sampling machine $S^?$, there exists a deterministic polynomial-time errorless heuristic oracle machine with a failure probability of at most n^{-2} for the distributional Σ_d^p problem $(L_M^{\mathcal{O}}, \mathcal{D}_S^{\mathcal{O}})$, defined as follows: $(\mathcal{D}_S^{\mathcal{O}})_n \equiv S^{\mathcal{O}}(1^n)$ for each $n \in \mathbb{N}$ and*

$$L_M^{\mathcal{O}} = \{x \in \{0,1\}^* : \exists w_1 \in \{0,1\}^{|x|} \forall w_2 \in \{0,1\}^{|x|}, \ldots, Q_d w_d \in \{0,1\}^{|x|}, M^{\mathcal{O}}(x, w_1, w_2, \ldots, w_d)\},$$

where $Q_d = \exists$ if d is an odd number; otherwise, $Q_d = \forall$.

By a simple padding argument on the instance size and the argument in [Imp95, Proposition 3], we obtain the following corollary to Theorem 9.1.25.

Corollary 9.1.26. *Let $\epsilon(n)$, c, and d denote the same parameters as in Theorem 9.1.25. With probability 1 over the choice of $\mathcal{O} := \mathcal{O}_{\epsilon,c,d}$, the event $\mathsf{Dist}\Sigma_d^{p\mathcal{O}} \subseteq \mathsf{AvgP}^{\mathcal{O}}$ occurs.*

Theorem 9.1.17 immediately follows from Theorem 9.1.20 and Corollary 9.1.26 by selecting $\epsilon(n) = 1/a(n)$ and sufficiently large c for ϵ^{-1} and d.

Proof of Theorem 9.1.25. For each n, let T_n be the maximum value of n^{2^c}, the 2^c-th power of the time for $S^?$ to generate an instance of size n, and the 2^c-th power of the time to execute $M^?$ on input of size n. Let $i_n = c^{-1} \log\log T_n$.

Now, we construct an errorless heuristic algorithm $B^{\mathcal{O}}$ that is given $x \in \{0,1\}^n$ as input and returns a value of $\mathcal{A}(\langle M, d\rangle, x, 1^{T_n^{2^c}})$. Note that $B^{\mathcal{O}}(x) = L_M^{\mathcal{O}}(x)$ unless $\mathcal{A}(\langle M, d\rangle, x, 1^{T_n^{2^c}})$ outputs "$\perp$". Thus, we will show the inequality

$$P_{n,M,S} := \Pr_{\mathcal{O},S}\left[\mathcal{A}(\langle M, d\rangle, x, 1^{T_n^{2^c}}) = \text{“}\perp\text{”} \text{ where } x \leftarrow S^{\mathcal{O}}(1^n)\right] \le O\left(\frac{1}{n^4}\right). \tag{9.8}$$

Then, by applying Markov's inequality, we have

$$\Pr_{\mathcal{O}}\left[\Pr_S\left[B^{\mathcal{O}}(x) = L_M^{\mathcal{O}}(x) \text{ where } x \leftarrow S^{\mathcal{O}}(1^n)\right] \ge \frac{1}{n^2}\right] \le O\left(\frac{1}{n^2}\right),$$

and the theorem follows from the Borel–Cantelli lemma and the countability of (M, S).

To show Eq. (9.8), we first show that the instance $x \in \{0,1\}^n$ is determined only by $\rho_{n',j}$ for $n' \le T_n$ and $j \le i_n - 1$ with a probability of at least $1 - O(n^{-4})$. Then, we will show that $\mathcal{A}(M, x, 1^{T_n^{2^c}})$ returns $M^{\mathcal{O}}(x)$ with a probability of at least $1 - O(n^{-4})$ under the condition that x is determined only by $\rho_{n',j}$ for $n' \le T_n$ and $j \le i_n - 1$.

By the same argument as the proof of Theorem 9.1.14, the first probability is bounded above by

$$\begin{aligned} T_n^{1/2^c} O\left(\max_{t^{-1}(T_n^{1/2^c})\le n'\le T_n^{1/2^c}} \{p(n'), q(n')\}\right) &= O\left(T_n^{1/2^c} q(t^{-1}(T_n^{1/2^c}))\right) \\ &= O\left(T_n^{1/2^c} \cdot (T_n^{1/2^c})^{-3(d+2)}\right) \\ &= O((T_n^{1/2^c})^{-(3d+5)}) \\ &= O(n^{-4}), \end{aligned}$$

where the last equation holds because $T_n \ge n^{2^c}$.

By the same argument as the proof of Theorem 9.1.14, the second probability that $\mathcal{A}$ returns "$\perp$" under the condition that the given instance $x \in \{0,1\}^n$ is determined only by $\rho_{n',j}$ for $n' \le T_n$ and $j \le i_n - 1$ is at most

$$\begin{aligned} O\left(T_n^2 \max_{t^{-1}(T_n)\le n'\le T_n} \{p(n')^{1/(d+2)}\ell(n')^2, q(n')^{1/(d+2)} \cdot 2^2\}\right) &= O\left(T_n^2 \max_{t^{-1}(T_n)\le n'\le T_n} t(n')^{-3}\right) \\ &= O(T_n^{-1}) \\ &= O(n^{-4}), \end{aligned}$$

where the last equation holds because $T_n \ge n^{2^c} \ge n^4$. □

Oracle Separation between Learning and Distributional PH

Theorem 9.1.27. *For any function ϵ such that $\omega(1) \le \epsilon(n) \le n/\omega(\log^2 n)$ and an arbitrary small constant $\delta \in (0,1)$, there exists an oracle $\mathcal{O}$ such that (1) $\mathsf{DistPH}^{\mathcal{O}} \subseteq \mathsf{AvgP}^{\mathcal{O}}$ and (2) $\mathsf{SIZE}^{\mathcal{O}}[n]$ is not weakly PAC learnable with membership queries in time $2^{O(\frac{n}{\epsilon(n)\log n})}$ on $\mathcal{D}$, where $\mathcal{D}$ is an arbitrary class of example distributions such that $\mathcal{D}_n$ contains all uniform distributions over subsets $S \subseteq \{0,1\}^n$ with $|S| \ge 2^{(1-\epsilon(n)^{-(1-\delta)})\cdot n}$.*

Proof sketch. Let $c = 3$. We construct an oracle $\mathcal{O}$, as in Section 9.1.4, where we set the parameters as follows:

$$t(n) = 2^{\frac{n}{\epsilon(n)\cdot\log n}},\ p(n) = t(n)^{-\epsilon(n)^\delta},\ \ell(n) = t(n)^2, q(n) = t(n)^{-\epsilon(n)^\delta},\ \text{and}\ i_{\max}(n) = c^{-1}\log\log t(n).$$

Then, the hardness of learning follows by the same argument as the proof of Theorem 9.1.20, where we can select the lower bound of G_z^* for a hard index z as

$$|G_z^*| \ge 2^{n-1} \cdot q(n)^{i_{\max}(n)} \ge 2^{(1-\frac{1}{3\epsilon(n)^{1-\delta}})n+1} \ge 2^{(1-\epsilon(n)^{-(1-\delta)})n},$$

for sufficiently large n. The average-case easiness of DistPH also holds by essentially the same argument as the proofs of Theorems 9.1.14 and 9.1.25. □

Proof of Lemma 9.1.21

Proof of Lemma 9.1.21. We can identify a random choice of n elements from U with n consecutive random choices of one element from U, where the chosen element is removed from U. We remark that these n choices are dependent on the previous choices, and we cannot directly apply the Chernoff bound. Instead, we apply the martingale theory. The basic background can be founded elsewhere [MU05].

For each $i \in \mathbb{N}$, let X_i be a random variable that returns 1 if the i-th chosen element is contained in S and 0 otherwise. Let $m = \sum_{i=1}^{n} X_i$. Then, the statement in the lemma is written as follows:

$$\Pr_{X_1,\ldots,X_n}\left[\left|m - \frac{M}{N}n\right| > \gamma \cdot \frac{M}{N}n\right] < 2e^{-2\gamma^2\cdot(\frac{M}{N})^2\cdot n}.$$

For each $i \in \mathbb{N} \cup \{0\}$, we define Z_i as

$$Z_i = \frac{M - \sum_{k=1}^{i} X_k}{N-i}n.$$

First, we show that these $Z_0, Z_1, \ldots, Z_n$ constitute a martingale.

Claim 9.1.28. *The sequence of $Z_0, Z_1, \ldots, Z_n$ is a martingale with respect to $X_1, \ldots, X_n$.*

Proof. It is sufficient to show that for each i, $\mathbb{E}[Z_{i+1}|X_1, \ldots, X_i] = Z_i$.

Fix $X_1, \ldots, X_{i-1}$ arbitrarily, where $i \leq n$. Let $X = \sum_{k=1}^{i} X_k$. If $X = M$, then $\mathbb{E}[Z_{i+1}|X_1, \ldots, X_i] = 0 = Z_i$. Even when $X < M$, the same equation holds as follows:

$$\begin{aligned}
\mathbb{E}[Z_{i+1}|X_1, \ldots, X_i] &= n \cdot \left[\frac{M-X-1}{N-i-1} \cdot \Pr[X_{i+1} = 1|X] + \frac{M-X}{N-i-1} \cdot \Pr[X_{i+1} = 0|X]\right] \\
&= n \cdot \left[\frac{M-X-1}{N-i-1} \cdot \frac{M-X}{N-i} + \frac{M-X}{N-i-1} \cdot \frac{(N-M)-(i-X)}{N-i}\right] \\
&= n \cdot \frac{(M-X)(N-i-1)}{(N-i-1)(N-i)} \\
&= n \cdot \frac{M-X}{N-i} \\
&= Z_i.
\end{aligned}$$

◇

Thus, the sequence of $Z_0, Z_1, \ldots, Z_n$ is a martingale (with respect to themselves).

For each $i \leq n$, under the condition on $X_1, \ldots, X_{i-1}$, we have

$$\frac{M - \sum_{k=1}^{i-1} X_k - 1}{N-i}n \leq Z_i \leq \frac{M - \sum_{k=1}^{i-1} X_k}{N-i}n.$$

Thus, we can show that

$$\begin{aligned}
Z_i - Z_{i-1} &\leq \frac{M - \sum_{k=1}^{i-1} X_k}{N-i}n - \frac{M - \sum_{k=1}^{i-1} X_k}{N-i+1}n \\
&= \frac{M - \sum_{k=1}^{i-1} X_k}{(N-i)(N-i+1)}n,
\end{aligned}$$

and

$$\begin{aligned} Z_i - Z_{i-1} &\geq \frac{M - \sum_{k=1}^{i-1} X_k - 1}{N-i} n - \frac{M - \sum_{k=1}^{i-1} X_k}{N-i+1} n \\ &= \frac{M - \sum_{k=1}^{i-1} X_k - (N-i+1)}{(N-i)(N-i+1)} n \end{aligned}$$

Therefore, if we define a new random variable B_i as

$$B_i = \frac{M - \sum_{k=1}^{i-1} X_k - (N-i+1)}{(N-i)(N-i+1)} n,$$

then we get

$$B_i \leq Z_i - Z_{i-1} \leq B_i + \frac{n}{N-i}.$$

Now, we apply the Azuma-Hoeffding inequality for $Z_0, \ldots, Z_n$. Then, for any real value $\lambda \geq 0$, we have

$$\begin{aligned} \Pr\left[|Z_n - Z_0| > \lambda\right] &< 2\exp\left(-\frac{2\lambda^2}{\sum_{i=1}^{n} (\frac{n}{N-i})^2}\right) \\ &\leq 2\exp\left(-\frac{2\lambda^2 (N-n)^2}{n^3}\right). \end{aligned}$$

Note that $Z_n = \frac{M - \sum_{i=1}^{n} X_i}{N-n} n = \frac{M-m}{N-n} n$ and $Z_0 = \frac{M}{N} n$. If we assume that $|m - \frac{M}{N} n| > \gamma \frac{M}{N} n$, then it is not hard to verify that

$$|Z_n - Z_0| > \gamma \cdot \frac{Mn^2}{N(N-n)}.$$

Therefore, by applying the above-mentioned inequality for $\lambda = \gamma \cdot \frac{Mn^2}{N(N-n)}$, we conclude that

$$\begin{aligned} \Pr\left[\left|m - \frac{M}{N} n\right| > \gamma \cdot \frac{M}{N} n\right] &\leq \Pr\left[|Z_n - Z_0| > \gamma \cdot \frac{Mn^2}{N(N-n)}\right] \\ &< 2\exp\left(-\frac{2(\frac{\gamma M n^2}{N(N-n)})^2 (N-n)^2}{n^3}\right) \\ &= 2\exp\left(-2\gamma^2 \cdot (\frac{M}{N})^2 \cdot n\right). \end{aligned}$$

□

9.2 Errorless vs. Error-Prone Average-Case Complexity

In this section, we study the difference between the two definition of "average-case easiness," *errorless* and *error-prone*[1] average-case easiness, through the lens of relativization. Remember that,

[1] It is originally called the "heuristic" complexity, and the term "error-prone" is due to the follow-up work [HS22].

in both formulations, an efficient algorithm needs to output a correct answer with high probability over a choice of random instances sampled from distribution $\mathcal{D}$. The difference is in the requirement when the algorithm cannot solve an instance. In the errorless setting, the algorithm is not allowed to output a wrong answer; instead, it is allowed to output a special symbol $\perp$, which represents the failure of the algorithm. In the error-prone setting, an algorithm is allowed to output a wrong answer, provided that the error probability of the algorithm is small. Below, we present further background on these formulations.

The difference between the two notions originates from two different motivations of studying average-case complexity. On one hand, Levin [Lev86] laid the foundation of the theory of average-case complexity of NP and introduced the notion of *average-case polynomial-time*, which is equivalent to errorless heuristic schemes [Imp95; BT06a]. The motivation of Levin is to clarify which distributional NP problems are hard, as some NP-complete problems are indeed easy on average with respect to natural distributions. Levin proved the distributional NP-completeness of a problem called the tiling problem. Although Levin's theory is applicable to both of the average-case notions, it is more natural to consider the notion of errorless average-case easiness in this context: Practical heuristic algorithms, such as SAT solvers, can be considered as errorless heuristics. A SAT solver is usually guaranteed to output the correct answer if it halts, but the solver may "fail" on some instances, i.e., may require a long time to halt on some instances. Levin's theory demonstrates that some distributional NP problems are hard and are unlikely to be solved by such heuristic algorithms. On the other hand, the errorless notion is not (necessarily) appropriate for discussing the security of cryptographic primitives. The foundational work of Blum and Micali [BM84] and Yao [Yao82] demonstrated that error-prone average-case hardness of some distributional NP problems is useful to build cryptographic primitives. Closing the gap between the errorless and error-prone average-case notions would unify the two motivations of studying average-case complexity. In his influential paper, Impagliazzo [Imp95] explicitly raised this question as an important research direction. The question can be formally stated as follows.

Question 9.2.1. *Is* $\mathsf{DistNP} \subseteq \mathsf{HeurP}$ *equivalent to* $\mathsf{DistNP} \subseteq \mathsf{AvgP}$*?*

Here, AvgP (resp. HeurP) denotes the class of distributional problems solvable on average by a polynomial-time algorithm in the errorless (resp. error-prone) setting; see Section 2.2 for a formal definition. We remark that DistNP denotes the class of distributional NP problems, i.e., $\mathsf{DistNP} = \{(L, \mathcal{D}) : L \in \mathsf{NP} \text{ and } \mathcal{D} \in \mathrm{PSAMP}\}$.

Giving an affirmative answer to Question 9.2.1 is necessary for basing the security of cryptography on the worst-case hardness of NP. An additional motivation was recently provided by Hirahara and Santhanam [HS22]: they identified a deep connection between the question of errorless versus error-prone average-case complexities and the question of constructing an instance checker for NP, which is another long-standing and important open question raised in the seminal work of Blum and Kannan [BK95].

Despite its importance, there does not seem to be an effective method for addressing this question, so it is natural to ask whether there is a technical barrier. This meta-approach is often considered in computational complexity theory and is useful for excluding hopeless proof techniques from consideration. For example, proof techniques that are captured by standard frameworks, such as relativization [BGS75], natural proofs [RR97], and algebrization [AW09], are known to be incapable of resolving the P versus NP question. However, to the best of our knowledge, there is no barrier for the question of errorless versus error-prone average-case complexities. In fact, Im-

pagliazzo [Imp95; Imp11] raised the open question of presenting a relativization barrier to Question 9.2.1.

Question 9.2.2. *Is there an oracle $\mathcal{O}$ such that* $\mathsf{DistNP}^{\mathcal{O}} \not\subseteq \mathsf{AvgP}^{\mathcal{O}}$ *and* $\mathsf{DistNP}^{\mathcal{O}} \subseteq \mathsf{HeurP}^{\mathcal{O}}$*?*

One of the main contributions of this section is to resolve this decade-old open question affirmatively. Before presenting the details of our results, we review the recent progress in complexity theory that demonstrates the notable power of the errorless average-case easiness of NP by *relativizing* proof techniques. Along the way, we provide additional questions related to errorless versus error-prone average-case complexities. We refer to the possible world in which $\mathsf{DistNP} \subseteq \mathsf{AvgP}$ (resp. $\mathsf{DistNP} \subseteq \mathsf{HeurP}$) but $\mathsf{P} \neq \mathsf{NP}$ as *errorless Heuristica* (resp. *error-prone Heuristica*). In any relativized errorless Heuristica, the following computational tasks regarding worst-case complexity are proved to be feasible.

Errorless Heuristica I: Approximating Complexity (Meta-Complexity) Meta-complexity is a field that studies the computational complexity of determining computational complexity. One central meta-computational problem is MINKT, which is the problem of determining the t-time-bounded Kolmogorov complexity of x for given $x \in \{0,1\}^*$ and $t \in \mathbb{N}$. Another well-studied problem is MCSP; for an input $x \in \{0,1\}^{2^n}$ (regarded as the truth table of a function), MCSP is the problem of determining the minimum size of the n-input circuit whose truth table corresponds to x, i.e., the circuit complexity of x.

Hirahara [Hir18] revealed that the approximation versions of the aforementioned problems are efficiently solvable in the *worst case* based on the errorless average-case easiness. Hirahara's theorem is stated as follows.

Theorem 9.2.3 ([Hir18])**.** *If* $\mathsf{DistNP} \subseteq \mathsf{AvgP}$*, then there exist a function* $\sigma(s,n) = \sqrt{s} \cdot \mathsf{polylog}(n)$ *and a constant* $\epsilon > 0$ *such that* $\mathrm{Gap}_{\sigma}\mathrm{MINKT} \in \text{pr-}\mathsf{ZPP}$ *and* $\mathrm{Gap}_{\epsilon}\mathrm{MCSP} \in \text{pr-}\mathsf{BPP}$*. Furthermore, these results are relativized.*[2]

Errorless Heuristica II: PAC Learning In Chapter 3, we proved that the worst-case requirements in learning are performed based on only the average-case easiness of NP under a natural computational assumption on example distributions.

Theorem 9.2.4 (3.1.1, simplified)**.** *If* $\mathsf{DistNP} \subseteq \mathsf{AvgP}$*, then* $\mathsf{P/poly}$ *is PAC learnable in polynomial time on all unknown* $\mathsf{P/poly}$*-samplable example distributions. Furthermore, this result is relativized.*

For simplicity, in this section we fix the confidence error to $1/3$ (i.e., the learner outputs a good hypothesis with probability at least $2/3$ over the choice of samples and randomness for the learner) because the confidence parameter is easily boosted by the standard repetition technique [HKLW88].

Errorless Heuristica III: No Auxiliary-Input Cryptography The aforementioned results are sufficient to break the security of any efficiently computable *auxiliary-input* cryptographic primitive, as observed in [ABX08; HS17], which is yet another notable consequence of $\mathsf{DistNP} \subseteq \mathsf{AvgP}$ because an adversary for an auxiliary-input primitive needs to succeed in breaking *all* primitives in the family, and this task is not captured directly as a distributional NP problem. Nevertheless, we can efficiently break any auxiliary-input cryptographic primitive in errorless Heuristica.

[2] A subsequent result [Hir20b] improved the approximation errors using a potentially non-relativizing proof technique of [BFP05].

Theorem 9.2.5. *If* $\mathsf{DistNP} \subseteq \mathsf{AvgP}$*, then there is no auxiliary-input one-way function. Furthermore, this result is relativized.*

The three theorems mentioned above demonstrate that several fascinating tasks concerning worst-case requirements can be performed in errorless Heuristica. By contrast, there is no result which shows the feasibility of a similar task in error-prone Heuristica. Thus, there are two possibilities: the errorless condition is essential in the aforementioned results, or they can be extended by similar (especially, relativizing) proof techniques. Determining which is correct is important to understand the capability and limitation of the technique for the worst-case to average-case reduction within NP developed by Hirahara [Hir18]. Particularly, a significant line of work [IL90; LP20; ACMTV21; LP21c; LP21a; IRS22; LP22] shows the characterization of a one-way function (OWF) based on the error-prone average-case hardness of several central problems in meta-complexity, including GapMINKT and GapMCSP. Therefore, if Hirahara's reduction can be extended to error-prone average-case analogues of these problems, then OWFs is characterized by the worst-case hardness of meta-computational problems. Despite many efforts, however, extending Hirahara's reduction is currently open. Proving Theorems 9.2.3, 9.2.4, and 9.2.5 in error-prone Heuristica is one natural and necessary approach for this research direction, where we consider the stronger assumption that $\mathsf{DistNP} \subseteq \mathsf{HeurP}$ (instead of the non-existence of OWFs) and attempt to solve easier problems such as breaking auxiliary-input cryptography.

Question 9.2.6. *Do Theorems 9.2.3, 9.2.4, and 9.2.5 also hold in error-prone Heuristica, i.e., under the assumption that* $\mathsf{DistNP} \subseteq \mathsf{HeurP}$*? Or, is there any barrier for such research directions?*

In this section, we address these questions and study the difference between the errorless average-case complexity and the error-prone average-case complexity from the perspective of relativization.

Our main contribution is the oracle construction for separating the error-prone average-case hardness and the errorless average-case hardness for distributional NP problems. Furthermore, the proposed oracle also separates the error-prone average-case hardness and (i) the hardness of approximating complexity (i.e., the lower bound of meta-complexity), (ii) the hardness of PAC learning, and (iii) the existence of auxiliary-input cryptographic primitives. Therefore, the proposed oracle exhibits the relativization barrier for Question 9.2.6.

We remark several points before presenting the result. When we consider the adversary defined as (a family of) circuits for some cryptographic primitives (e.g., auxiliary-input primitives and hitting set generators), we regard a size function $s(n)$ of an adversary as a function in the length of a hidden seed instead of output of the primitive for simplicity. In addition, we regard a time-bound function of a learning algorithm as a function in the length of examples, i.e., the input size to the target function. In this section, we consider GapMINKT with a relaxed (i.e., easier) requirement than one discussed in [Hir18; Hir20b], in the sense that we do not consider the time-bound in "no" cases. In other words, we only need to distinguish efficiently generated strings from strings no short program can generate even in time-unbounded settings. The relaxed formulation strengthens our separation result because it shows the hardness of such a relaxed version of GapMINKT.

Definition 9.2.7 (GapMINKT)**.** *For a function* $\sigma\colon \mathbb{N}\times\mathbb{N}\to\mathbb{N}$*,* $\mathrm{Gap}_\sigma\mathrm{MINKT}$ *is a promise problem* (Π_Y, Π_N) *defined as* $\Pi_Y = \left\{(x, 1^s, 1^t) : \mathrm{K}^t(x) \leq s\right\}$ *and* $\Pi_N = \left\{(x, 1^s, 1^t) : \mathrm{K}(x) > s + \sigma(s, |x|)\right\}$*.*

Now, we present the main theorem.

Theorem 9.2.8. *For any constant* $a > 0$*, there exists an oracle* $\mathcal{O}$ *relative to which the following hold:*

- **(Error-prone average-case easiness of NP)** $\mathsf{DistNP}^{\mathcal{O}} \subseteq \mathsf{HeurP}^{\mathcal{O}}$.
- **(Errorless average-case hardness of NP)** $\mathsf{DistNP}^{\mathcal{O}} \nsubseteq \mathsf{AvgSIZE}^{\mathcal{O}}[2^{an/\log n}]$.
- **(Lower bound of meta-complexity)** $\mathrm{Gap}_{\sigma}\mathrm{MINKT}^{\mathcal{O}} \notin \text{pr-}\mathsf{SIZE}^{\mathcal{O}}[2^{an/\log n}]$ *for any* $\sigma(s,n) = o(s)\cdot\mathsf{polylog}(n)$. *In addition, for each* $\epsilon \in [0,1]$, *there exists* $\delta \in (0,1)$ *such that* $\mathrm{Gap}_{\epsilon}\mathrm{MCSP}^{\mathcal{O}} \notin \text{pr-}\mathsf{SIZE}^{\mathcal{O}}[2^{n^{\delta}}]$.
- **(Worst-case hardness of learning on uniform distributions)** $\mathsf{SIZE}^{\mathcal{O}}[n]$ *is not weakly PAC learnable with membership queries (MQ) on the uniform distribution by nonuniform* $O(2^{an/\log n})$*-time algorithms. Furthermore, there exists a polynomial* $s(n)$ *such that* $\mathsf{SIZE}^{\mathcal{O}}[s(n)]$ *is not weakly PAC learnable with MQ on the uniform distribution by nonuniform* $2^n/n^{\omega(1)}$*-time algorithms.*
- **(Average-case hardness of distribution-free learning)** *There exists a polynomial* $s(n)$ *such that* $\mathsf{SIZE}^{\mathcal{O}}[s(n)]$ *is not weakly PAC learnable on average in the BFKL model with respect to all unknown example distribution and a fixed samplable distribution over* $\mathsf{SIZE}^{\mathcal{O}}[s(n)]$ *by nonuniform* $O(2^{an/\log n})$*-time algorithms. Furthermore,* $\mathsf{SIZE}^{\mathcal{O}}[n]$ *is not weakly PAC learnable on average in the BFKL model with respect to all unknown example distribution and a fixed samplable distribution over* $\mathsf{SIZE}^{\mathcal{O}}[n]$ *by nonuniform* $O(2^{n^{\epsilon}})$*-time algorithms for some constant* $\epsilon > 0$.
- **(Auxiliary-input cryptographic primitives)** *There exist a hitting set generator (HSG), an auxiliary-input one-way function (AIOWF), an auxiliary-input pseudorandom generator (AIPRG), and an auxiliary-input pseudorandom function (AIPRF) against* $\mathsf{SIZE}^{\mathcal{O}}[2^{an/\log n}]$.

The lower bound in the oracle separation is considerably stronger than the polynomial lower bound and holds for the nonuniform computation model. We remark that Theorem 9.2.8 shows strong evidence that profoundly new techniques are necessary for further improvements of the main results in Chapters 3 and 4, as discussed in Sections 3.1 and 4.3.5.

It is worthy of note that Wee [Wee06] constructed an oracle relative to which $\mathsf{DistNP} \nsubseteq \mathsf{HeurP}$, and no AIOWF exists against $\mathsf{P/poly}$, which is the opposite separation of one of our results. Combined with Wee's result, our results show that auxiliary-input cryptography and the error-prone average-case hardness of NP are incomparable by any relativizing proof.

Furthermore, since $\mathsf{DistNP}^{\mathcal{O}} \subseteq \mathsf{HeurP}^{\mathcal{O}}$ is sufficient to break any one-way function, we can immediately derive the strong oracle separation between OWF and subexponentially and nonuniformly secure AIOWF, which improves the previous oracle separation between OWF and (polynomially and uniformly secure) AIOWF in [Tre10; Nan21b].[3]

Corollary 9.2.9. *For any constant* $a > 0$, *there exists an oracle* $\mathcal{O}$ *relative to which the following hold:*

- **(Nonexistence of OWF)** *There exists no infinitely-often one-way functions.*
- **(Subexponentially and nonuniformly secure AIOWF)** *There exist an auxiliary-input one-way function secure against* $\mathsf{SIZE}^{\mathcal{O}}[2^{an/\log n}]$.

[3] In fact, a central idea in this section is not so much involved with the improvement in Corollary 9.2.9. The novelty of our result lies in improving the easiness part from the nonexistence of OWF to $\mathsf{DistNP} \subseteq \mathsf{HeurP}$.

9.2.1 Related Work

Other related oracle separation results are mentioned in Section 9.3. Hirahara and Santhanam [HS22] also addressed the errorless complexity versus error-prone complexity problem, and they showed that the equivalence between a non-adaptive errorless to error-prone reduction for NP and an average-case instance checker for NP. They also discussed Question 9.2.1 for other classes of distributional problems such as DistPH and Dist(UP ∩ coUP) and showed that Dist(UP ∩ coUP) ⊆ AvgP if and only if Dist(UP ∩ coUP) ⊆ HeurP, i.e., they resolved Question 9.2.1 for the subclass UP ∩ coUP of NP.

9.2.2 Proof Ideas

We present ideas behind our oracle separation. Again, the oracle construction is based on the one presented by Impagliazzo [Imp11], in which the worst-case hardness and the errorless average-case easiness are separated for NP. First, we briefly review the idea and subsequently present its adjustment for the separation between the errorless average-case hardness and the error-prone average-case hardness for NP. For simplicity, we only consider the uniform distribution as the distribution over instances (instead of all sampleable distributions) and a lower bound for P/poly (instead of $\mathsf{SIZE}[2^{an/\log n}]$) in this section.

Oracle Separation between NP ⊈ P/poly and DistNP ⊆ AvgP

We briefly review the oracle $\mathcal{O}$ in [Imp11] with a slight modification for applying the standard switching lemma (as discussed in Section 9.1). The oracle $\mathcal{O}$ consists of two oracles $\mathcal{V}$ and $\mathcal{A}$ and a hidden internal random function $f\colon \{0,1\}^n \to \{0,1\}^{m(n)}$, where $\mathcal{V}$ represents a verification oracle for the NP relation $R(x, f(x))$ which makes NP worst-case hard, and $\mathcal{A}$ represents a restrictive NP oracle which makes NP average-case easy while retaining the worst-case hardness. The NP oracle $\mathcal{A}$ is given a description of a nondeterministic oracle machine M, an input x, and a time bound T (of the form 1^{T^4} to prevent the circular call for $\mathcal{A}$), simulates $M^{\mathcal{V},\mathcal{A}}(x)$ in T time, and returns the answer, where we allow $\mathcal{A}$ to use only partial values of f on randomly selected positions. If the execution is determined only by the partial information, then $\mathcal{A}$ returns the result; otherwise, $\mathcal{A}$ returns $\perp$. In the actual construction, we introduce a structure in f by multiple applications of random restrictions in the selection of f to address the issue on query access to $\mathcal{A}$ by M. Then, for a given time bound 1^{T^4}, we only apply from the first to $i_T := 2^{-1}\log\log T$-th random restrictions.

The average-case easiness of NP follows from the switching lemma for DNFs, where we regard each $f(y)_i$ as a binary variable for each input y and position i (assigned in the random selection of f) and $M^{\mathcal{V},\mathcal{A}}(x)$ (executed in T time) as a $m(n)\cdot T$-DNF formula. By contrast, the worst-case hardness is shown by considering the $\mathsf{NP}^{\mathcal{V},\mathcal{A}}$ problem $L = \{\langle x, i\rangle : \exists y \text{ s.t. } \mathcal{V}(x,y) = 1 \text{ and } y_i = 1\}$ (remember that L is indeed in $\mathsf{UP}^{\mathcal{V},\mathcal{A}} \cap \mathsf{coUP}^{\mathcal{V},\mathcal{A}}$). Particularly, any polynomial-size circuit C can only access up to the $2^{-1}\log\log \mathsf{poly}(n) = O(\log\log n)$-th random restriction. Therefore, if there still remain many unassigned values in the $O(\log\log n)$-th random restriction, then C should guess such values at random to find the witness $f(x)$ for L, which implies the worst-case hardness.

The aforementioned oracle yields the *errorless* average-case easiness because $\mathcal{A}$ returns $\perp$ in the case in which the simulation of the given nondeterministic machine is not determined by the random restrictions. Therefore, a natural idea to separate the errorless and error-prone complexities is that we make $\mathcal{A}$ return a wrong answer in such cases. To implement this idea, the following concerns should be addressed. First, how should the answer from $\mathcal{A}$ be determined in such cases? Note that

$\mathcal{A}$ cannot use the values of f assigned at higher levels in the structure to identify the wrong answer because it causes a circular problem, i.e., the DNF representing $M^{\mathcal{V},\mathcal{A}}(x)$ is not determined only by up to the $(i_T - 1)$-th random restriction anymore. Second, how should a distributional problem be determined for the errorless average-case hardness? Particularly, Hirahara and Santhanam [HS22] showed the equivalence between the errorless average-case easiness and the error-prone average-case easiness of $\mathsf{UP} \cap \mathsf{coUP}$ by relativizing proof techniques. Thus, we cannot hope to prove the errorless average-case hardness for the same $\mathsf{UP} \cap \mathsf{coUP}$ problem L under the error-prone average-case easiness of NP.

First Attempt for $\mathsf{DistNP} \nsubseteq \mathsf{AvgP/poly}$ and $\mathsf{DistNP} \subseteq \mathsf{HeurP}$

The answer to the first question is relatively simple: we make $\mathcal{A}$ always answer 0. The intuition behind this is that an oracle machine given 1 as an answer from $\mathcal{A}$ (for some NP-type statement) can also obtain the witness for this assertion by the self-reducibility of NP; otherwise, the oracle machine can detect the error of $\mathcal{A}$ and output $\perp$. Thus, any error-prone algorithm can be translated into an errorless algorithm when $\mathcal{A}$ answers 1 as a wrong answer at some stage. By contrast, if $\mathcal{A}$ answers 0, i.e., declares "no witness," then there seems no efficient way to detect this error. Thus, we let $\mathcal{A}$ always answer 0, and this choice is indeed crucial in the proof.

By contrast, the answer to the second question is less obvious. Our approach is to construct a hitting set generator (HSG) instead of determining a distributional problem directly. A HSG (against $\mathsf{P/poly}$) is a (family of) efficiently computable function $G\colon \{0,1\}^n \to \{0,1\}^{m(n)}$ which stretches the seed (i.e., $m(n) > n$) and hits any language recognized by a polynomial-size circuit. Specifically, if a polynomial-size circuit C accepts more than half of the strings in $\{0,1\}^{m(n)}$, then C also accepts $G(x)$ for some $x \in \{0,1\}^n$ (for infinitely many $n \in \mathbb{N}$). Constructing a HSG for the errorless average-case hardness is a natural approach because it immediately yields a natural distributional NP problem $(\mathrm{Im}G, \mathrm{Uniform})$ that is hard on average in the errorless setting, and Hirahara [Hir20a] demonstrated the equivalence between the errorless average-case hardness of PH and the existence of PH-computable HSGs.

A first attempt to construct a HSG is that we regard the random function $f\colon \{0,1\}^n \to \{0,1\}^{m(n)}$ as a generator, where we let $m(n) > n$. Now, we replace the verification oracle $\mathcal{V}$ with $\mathcal{F}$ defined as $\mathcal{F}(x,i) = f(x)_i$ because the generator requires direct access to f for computing its values. Then, we define the candidate $G^{\mathcal{F},\mathcal{A}}\colon \{0,1\}^n \to \{0,1\}^{m(n)}$ for a HSG as $G^{\mathcal{F},\mathcal{A}}(x) = \mathcal{F}(x,1) \circ \cdots \circ \mathcal{F}(x,m(n))\,(= f(x))$. However, this generator G is not a HSG, and G can be broken efficiently by using the partial information of f efficiently obtained from $\mathcal{A}$, informally as follows: For each random restriction, an expected fraction of unassigned values in f is $n^{-\omega(1)}$. Thus, for a given string $y \in \{0,1\}^{m(n)}$, we can easily detect the case of $y = G(x)$ for a large fraction of $x \in \{0,1\}^n$ by asking an NP-type query to $\mathcal{A}$ such as "Is there $x \in \{0,1\}^n$ such that $G(x)$ is *partially* consistent with y?" because the answer tends to be fixed to 1 only by the random restriction in $\mathcal{A}$ if such an x exists. After applying the random restrictions $\omega(1)$ times, the aforementioned strategy is sufficient for detecting all the cases of $y \in \mathrm{Im}G$. Thus, some different approach is required.

We remark that we can now regard executing a nondeterministic $M^{\mathcal{F},\mathcal{A}}(x)$ in T time as a T-DNF formula (instead of an $m(n) \cdot t$-DNF) because $\mathcal{F}$ accesses only one entry in f for each query.

Our Construction: Random Restriction with Masks

To construct a HSG, we introduce a new type of random restrictions, *random restriction with masks*, which is crucial to solve Question 9.2.2. A random restriction with masks to $f\colon \{0,1\}^n \to \{0,1\}^{m(n)}$ with parameter $p \in [0,1]$ (i.e., the unset probability) is performed as follows: First, we select a random subset $S_1 \subseteq \{0,1\}^n$ of size $p \cdot 2^n$ and then apply a standard random restriction with unset probability p to a variable set $\{f(x)_i : x \in \{0,1\}^n \setminus S_1 \text{ and } i \in [m(n)]\}$, i.e., the random set S_1 performs as a "mask" that prevents restriction. This variant of random restriction is extended to multiple applications inductively as follows: Let S_i be the random subset (i.e., the mask) selected in the i-th random restriction with masks to f. Next, the $(i+1)$-th restriction (with parameter p) is performed by selecting a random subset $S_{i+1} \subseteq S_i$ of size $p \cdot |S_i|$ and applying random restriction to variables except for S_{i+1}.

We consider a modified oracle in which the oracle construction is the same as previously mentioned except that we apply random restrictions with masks instead of the standard random restrictions. For now, we select the unset probability $p(n) = n^{-\log n}$. This choice is sufficient for a HSG against $\mathsf{P/poly}$ and the statement that $\mathsf{DistNP} \not\subseteq \mathsf{AvgP/poly}$. Note that $p(n)$ should be selected more carefully according to the size complexity of the adversary in the formal argument (for the detail, see Section 9.2.3).

Specifically, we randomly select the aforementioned oracles $\mathcal{F}$ and $\mathcal{A}$ by selecting the internal random function $f\colon \{0,1\}^n \to \{0,1\}^{m(n)}$ with $\log n$ applications of random restrictions with masks for each $n \in \mathbb{N}$ (after applying random restrictions, we also select the remaining values of f at random). For each $n \in \mathbb{N}$, let $S_{n,\log n} \subseteq \{0,1\}^n$ be the random mask selected in the $\log n$-th restriction. Then, we have $|S_{n,\log n}| = p(n)^{\log n} \cdot 2^n = 2^{n-(\log n)^3}$. Thus, there exist exponentially many $z \in S_{n,\log n} \subseteq \{0,1\}^n$ (we call these hard indices) such that no value in $f(z)$ is assigned by the $\log n$-th restriction. Remember that for a query $(M, x, 1^{T^4})$, the oracle $\mathcal{A}$ applies only up to the $i_T := 2^{-1} \log\log T$-th random restriction. Since any polynomial-size adversary C can make a query only with $T = \mathsf{poly}(n)$, C can only access up to the $O(\log\log n)$-th restrictions. Therefore, any polynomial-size adversary cannot obtain any information about $f(z)$ from $\mathcal{A}$ for each hard index z, and oracle access to $\mathcal{F}(z,i) = f(z)_i$ is indistinguishable from access to a random function for such adversaries.

The aforementioned argument is sufficient for constructing a HSG. In fact, by defining the generator G as $G^{\mathcal{F},\mathcal{A}}(x) = \mathcal{F}(x,1) \circ \cdots \circ \mathcal{F}(x,m(n))$, we can show that G is a HSG against $\mathsf{P/poly}$ by a similar argument as in Section 9.1. Furthermore, the random restriction method with masks has another advantage: even if we select exponentially large $m(n)$, it still provides hard indices z such that $\mathcal{A}$ does not reveal any information of $f(z)$ to polynomial-size adversaries. Specifically, by letting $m(n) = 2^n \cdot n$ (i.e., the length of the truth table of a mapping from n-bit to n-bit), we can prepare an auxiliary-input oracle $\mathcal{F}\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ such that $\mathcal{F}(z,\cdot)$ is (computationally) indistinguishable from a random oracle for subexponentially many hard z's. Therefore, $F(\cdot,\cdot)$ is an AIOWF because it is known that a random oracle is also a OWF with probability 1 over the choice of the random oracle [cf. IR89; GT00]. Furthermore, by the technique presented in [Zim98], we can construct an AIPRG based on the auxiliary-input analog of a random oracle with less security loss than the general methods to convert OWFs into PRGs [e.g. HILL99]. In the formal proof of Theorem 9.2.8, we first construct such an AIPRG and subsequently show the related hardness notion (e.g., HSGs and the hardness of learning) to prevent security loss. Note that the aforementioned argument does not yield standard cryptographic primitives such as OWFs because the set of hard indices is selected at random, and there is no efficient sampling algorithm

that selects a hard index with high probability.

A random restriction with masks assigns fewer variables than a standard random restriction. Therefore, the remaining problem is whether the error-prone average-case easiness is preserved in the modified oracle construction. This issue can be addressed by the choice of the answer (i.e., 0) from $\mathcal{A}$ when the simulation is not determined by random restrictions. The proof is outlined as follows.

For convenience, we regard that a random restriction with masks is performed as follows: (i) a standard random restriction is applied to remaining variables at the stage, (ii) the random subset $S_{i+1} \subseteq S_i$ is selected in the same manner, and (iii) the values in $f(z)$ are returned to unassigned for each $z \in S_{i+1}$. Let us call the first step (resp. the second and third steps) a *restriction* (resp. *reverse*) step. The random restriction in the restriction step is merely a standard one. Thus, by the standard switching lemma, we can show that the value of a T-DNF ϕ (representing the execution of a nondeterministic machine in T time) is determined with high probability at this stage. Therefore, it is sufficient to show that the answer from $\mathcal{A}$ rarely changes in the reverse step.

For simplicity, we use the notation $*$ to refer to the cases in which the DNF ϕ is not fixed by the random restriction. Then, there are $3 \times 3 = 9$ possibilities about the change in the state on the restricted ϕ, i.e., from $\{0, 1, *\}$ (in the restriction step) to $\{0, 1, *\}$ (in the reverse step). Obviously, we do not need to consider the following 3 cases: $\{0\} \to \{0\}$, $\{1\} \to \{1\}$, and $\{*\} \to \{*\}$. Since we cancel some assignments in the reverse step, the following 4 cases do not occur: $\{*\} \to \{0, 1\}$, $\{0\} \to \{1\}$, and $\{1\} \to \{0\}$. Furthermore, because $\mathcal{A}$ answers 0 in the case of $*$, we do not need to consider the case of $\{0\} \to \{*\}$. Therefore, the remaining case is only $\{1\} \to \{*\}$.

We show that the case of $\{1\} \to \{*\}$ rarely occurs as follows. Since the T-DNF formula ϕ is satisfied in the restriction step, there must exist a satisfied term τ of size T. If ϕ becomes unfixed in the reverse step, then τ is also unfixed. This event occurs only if there exists $z \in \{0, 1\}^n$ such that some variable $f(z)_i$ is contained in τ (for some i), and z is selected on the choice of the random subset in the reverse step. Since τ covers at most T indices z in literals, this probability is at most $T \cdot p(n) = T \cdot n^{-\log n}$. Particularly, for solving a NP problem by $\mathcal{A}$, we only need to simulate a nondeterministic machine in $\mathsf{poly}(n)$ times, so we can let $T = \mathsf{poly}(n)$. Therefore, the error probability that the answer from $\mathcal{A}$ is changed in the reverse step is negligible.

Based on the ideas above, we complete the proof of Theorem 9.2.8 in the subsequent sections. In Section 9.2.3, we present the oracle construction. In Section 9.2.4, we show the average-case easiness of NP in the error-prone setting. In Section 9.2.5, we show the existence of AIPRG and the other hardness statements, including the average-case hardness of NP in the errorless setting and the hardness of learning.

9.2.3 Oracle Construction

In this section, we formally present the oracle construction.

For each $p \in [0, 1]$ and set S of variables taking binary values, we define a p-random restriction ρ to S as a partial assignment $\rho\colon S \to \{0, 1, *\}$ (where $*$ represents "unassigned") randomly selected as follows: for each $x \in S$,

$$\rho(x) = \begin{cases} * & \text{with probability } p \\ 0 & \text{with probability } (1-p)/2 \\ 1 & \text{with probability } (1-p)/2. \end{cases}$$

For every restriction ρ to S and function f defined on S, we let $f|_\rho$ denote the restricted function obtained by applying a partial assignment to f according to ρ.

Let $a > 0$ be a parameter. The oracle construction is the following.

Construction. *$\mathcal{O}_a = \mathcal{F} + \mathcal{A}$, where each oracle is randomly selected according to the following process:*

1. *Let $t(n) = 2^{an/\log n}$ be the upper bound on the time of nonuniform adversaries and $c = 7a$.*
2. *Define functions p and $i_{\max}$ as $p(n) = t(n)^{-6}$ and $i_{\max}(n) = \frac{1}{c}\log\log t(n)$. Here, p is a parameter of random restriction, and $i_{\max}$ is the number of applications of random restrictions.*
3. *For each $n \in \mathbb{N}$, define a set $V_{n,0}$ of variables taking binary values as follows:*
$$V_{n,0} = \{F_{z,x,\ell} : z, x \in \{0,1\}^n, \ell \in [n]\}.$$
4. *For each $n \in \mathbb{N}$, let $S_{n,0} = \{0,1\}^n$.*
5. *For each $n \in \mathbb{N}$ and $i \in [i_{\max}(n) - 1]$, we inductively (on i) select a $p(n)$-random restriction $\rho^*_{n,i}$ to $V_{n,i-1}$ and a random subset $S_{n,i} \subseteq S_{n,i-1}$ of size $p(n) \cdot |S_{n,i-1}|$. Then, we define a restriction $\rho_{n,i}$ to $V_{n,i-1}$ and a subset $V_{n,i} \subseteq V_{n,i-1}$ as follows:*
$$\rho_{n,i}(z,x,\ell) = \begin{cases} * & \text{if } z \in S_{n,i} \\ \rho^*_{n,i}(z,x,\ell) & \text{otherwise} \end{cases}$$
$$V_{n,i} = \rho_{n,i}^{-1}(*) \left(= {\rho^*_{n,i}}^{-1}(*) \cup \{F_{z,x,\ell} : z \in S_{n,i}\}\right).$$
We also define $\rho_{n,i_{\max}(n)}$ as a full assignment to $V_{n,i_{\max}(n)-1}$ selected uniformly at random. Let $\rho_{n,i} \equiv \rho_{n,i_{\max}(n)}$ for each $i \geq i_{\max}(n) + 1$. We use the notation $\rho_{n,\leq i}$ to represent the composite restriction $\rho_{n,1} \cdots \rho_{n,i}$ to $V_{n,0}$ for each n and i.
6. *Define $\mathcal{F} = \{\mathcal{F}_n\}_{n\in\mathbb{N}}$, where $\mathcal{F}_n \colon \{0,1\}^n \times \{0,1\}^n \times [n] \to \{0,1\}$, as $\mathcal{F}_n(z,x,\ell) = \rho_{n,\leq i_{\max}(n)}(z,x,\ell)$.*
7. *Define $\mathcal{A}$ as follows: On input $(M, x, 1^{T^{2^c}})$, where M is a nondeterministic oracle machine, $x \in \{0,1\}^*$, and $T \in \mathbb{N}$, the oracle $\mathcal{A}(M, x, 1^{T^{2^c}})$ returns 0 or 1 according to the following procedure:*

   1: Let $i_T := \frac{1}{c}\log\log T$.
   2: Construct a T-DNF ϕ on variables in $V_{n,0}$ representing the execution of $M^{\mathcal{F}+\mathcal{A}}(x)$ in T steps, where the top-most OR corresponds to the nondeterminism on a possible choice of $\mathcal{F}$ (say, $\mathcal{F}'$) and an accepting path of $M^{\mathcal{F}'+\mathcal{A}}(x)$, and each term performs verification whether M's at most T queries (say, $(z_1,x_1,\ell_1),\ldots,(z_q,x_q,\ell_q)$ for some $q \leq T$) are consistent with the actual choices of $\mathcal{F}$, i.e., for each $i \in [q]$, the term contains F_{z_i,x_i,ℓ_i} if $\mathcal{F}'(z_i,x_i,\ell_i) = 1$; otherwise, $\neg F_{z_i,x_i,\ell_i}$ as a literal.
   *3: If $\phi|_{\rho_{1,\leq i_T},\ldots,\rho_{T,\leq i_T}} \equiv b$ for some $b \in \{0,1\}$, then **return** b, otherwise, **return** 0.*

We can verify that $\mathcal{A}$ is well-defined (i.e., not circular on recursive calls for $\mathcal{A}$) as follows.

Proposition 9.2.10. *For each input, the value of $\mathcal{A}(M, x, 1^{T^{2^c}})$ is determined only by $\rho_{n,j}$ (equivalently, $\rho^*_{n,j}$ and $S_{n,j}$) for $n \leq T$ and $j \leq i_T$ (remember that $i_T = \frac{1}{c}\log\log T$).*

Proof. We show the proposition by induction on T. Remember that, on input $(M, x, 1^{T^{2^c}})$, the oracle $\mathcal{A}$ first makes a T-DNF ϕ based on M independently of the values of $\mathcal{F}$.

Suppose that M makes some query $(M', x', 1^{T'^{2^c}})$ to $\mathcal{A}$ for constructing ϕ. Since the length of such a query is at most T, we have $T'^{2^c} \leq T$ and

$$i_{T'} = \frac{1}{c} \log \log T' \leq \frac{1}{c} \log \log T^{\frac{1}{2^c}} = \frac{1}{c} \log \log T - 1 = i_T - 1.$$

By the induction hypothesis, the answer of $\mathcal{A}(M', x', 1^{T'^{2^c}})$ is determined by only $\rho_{n,j}$ for $n \leq T'$ and $j \leq i_T - 1$, and so is ϕ. Then, $\mathcal{A}$ determines the answer by restricting ϕ by $\rho_{n,j}$ for $n \leq T$ and $j \leq i_T$. Therefore, $\mathcal{A}(M, x, 1^{T^{2^c}})$ is determined only by $\rho_{n,j}$ for $n \leq T$ and $j \leq i_T$. □

When the parameter a is clear from the context, we omit the subscript a from $\mathcal{O}_a$.

9.2.4 Error-Prone Average-Case Easiness of NP

In this section, we show the error-prone average-case easiness of NP.

Theorem 9.2.11. *With probability 1 over the choice of $\mathcal{O}_a$, $\mathsf{DistNP}^{\mathcal{O}_a} \subseteq \mathsf{HeurP}^{\mathcal{O}_a}$ holds.*

First, we introduce several notations. For each choice of $\mathcal{O}$, we define the oracle $\mathcal{A}^*$ in the same manner as $\mathcal{A}$ except we apply $\rho_{1,\leq i_T - 1}\rho^*_{1,i_T}, \ldots, \rho_{T,\leq i_T - 1}\rho^*_{T,i_T}$ to ϕ instead of $\rho_{1,\leq i_T}, \ldots, \rho_{T,\leq i_T}$. Note that $\mathcal{A}^*$ executes a given nondeterministic machine M with access to $\mathcal{A}$ (rather than $\mathcal{A}^*$) to construct the corresponding DNF ϕ. We can verify that $\mathcal{A}^*$ is well-defined (i.e., not circular) in the same manner as Proposition 9.2.10.

Now, we show that $\mathcal{A}$ and $\mathcal{A}^*$ do not differ considerably.

Lemma 9.2.12. *For each input $(M, x, 1^{T^{2^c}})$ to $\mathcal{A}$, we have that*

$$\Pr_{\mathcal{O}}\left[\mathcal{A}(M, x, 1^{T^{2^c}}) \neq \mathcal{A}^*(M, x, 1^{T^{2^c}})\right] = O(T^{-4}).$$

Proof. Let $i := i_T = (1/c) \log \log T$. For all $n \leq T$ and $j \leq i - 1$, we fix $\rho^*_{n,j}$, $S_{n,j}$, and $\rho^*_{n,i}$ arbitrarily; let C_T denote this condition. Notice that the DNF formula ϕ_{C_T} constructed by $\mathcal{A}$ and $\mathcal{A}^*$ is determined only by C_T because all answers to recursive calls for $\mathcal{A}$ are determined by C_T as in Proposition 9.2.10. Let $\phi'_{C_T} = \phi_{C_T}|_{\rho_{1,\leq i-1}, \ldots, \rho_{T,\leq i-1}}$. Then, the value of $\mathcal{A}(M, x, 1^{T^{2^c}})$ (resp. $\mathcal{A}^*(M, x, 1^{T^{2^c}})$) is determined by $\phi'_{C_T}|_{\rho_{1,i}, \ldots, \rho_{T,i}}$ (resp. $\phi'_{C_T}|_{\rho^*_{1,i}, \ldots, \rho^*_{T,i}}$).

For any DNF formula ϕ and a restriction ρ, there are the following three cases: (i) $\phi|_\rho \equiv 0$, (ii) $\phi|_\rho \equiv 1$, or (iii) $\phi|_\rho$ does not become a constant (we write this case as $\phi|_\rho \equiv *$). Following this case analysis, there exist $3^2 = 9$ cases on $(\phi'_{C_T}|_{\rho^*_{1,i}, \ldots, \rho^*_{T,i}}, \phi'_{C_T}|_{\rho_{1,i}, \ldots, \rho_{T,i}})$. However, since each $\rho_{n,i}$ is a subrestriction of $\rho^*_{n,i}$ (i.e., $\rho_{n,i}$ assigns values only to variables that are also assigned by $\rho^*_{n,i}$), the following 4 cases do not occur: $(0,1), (1,0), (*,0)$, and $(*,1)$. Further, in the cases of $(0,0), (1,1), (*,*)$, and $(*,0)$, the answers by $\mathcal{A}^*$ and $\mathcal{A}$ do not differ because $\mathcal{A}^*$ and $\mathcal{A}$ return 0 in the case of $*$. Thus, we only need to consider the case of $(1,*)$.

By the aforementioned argument, the probability in the lemma is expressed as follows:

$$\Pr_{\mathcal{O}}\left[\mathcal{A}(M, x, 1^{T^{2^c}}) \neq \mathcal{A}^*(M, x, 1^{T^{2^c}})\right] = \mathop{\mathbb{E}}_{C_T}\left[\Pr_{S_{1,i}, \ldots, S_{T,i}}\left[\phi'_{C_T}|_{\rho_{1,i}, \ldots, \rho_{T,i}} \equiv * \middle| C_T, \phi'_{C_T}|_{\rho^*_{1,i}, \ldots, \rho^*_{T,i}} \equiv 1\right]\right].$$

To bound the probability in the right-hand side for each condition, we consider the case in which $\phi'_{C_T}|_{\rho^*_{1,i},\dots,\rho^*_{T,i}} \equiv 1$. Then, we can select a term τ in ϕ'_{C_T} such that $\tau|_{\rho^*_{1,i},\dots,\rho^*_{T,i}} \equiv 1$. For each $n \le T$, define $Z_n \subseteq \{0,1\}^n$ as

$$Z_n = \{z \in \{0,1\}^n : \exists (x,\ell) \in \{0,1\}^n \times [n] \text{ s.t. a variable } F_{z,x,\ell} \text{ is contained in } \tau\}.$$

Since ϕ'_{C_T} is a T-DNF, $|Z_n| \le T$ for each $n \le T$. Furthermore, for any $n \in \mathbb{N}$ such that $i_{\max}(n) \le i-1$, we have that $Z_n = \emptyset$ because all such variables must be assigned by $\rho_{n,\le i-1}$. If $\phi'_{C_T}|_{\rho_{1,i},\dots,\rho_{T,i}} \equiv *$, then $\tau|_{\rho_{1,i},\dots,\rho_{T,i}} \equiv *$ must hold. This event occurs only if $\bigcup_{n\le T}(S_{n,i} \cap Z_n) \neq \emptyset$ holds. Since each $S_{n,i}$ is selected from $S_{n,i-1}$ uniformly at random, this probability is bounded above by

$$\begin{aligned}
\Pr\left[\bigcup_{n\le T}(S_{n,i}\cap Z_n) \neq \emptyset\right] &\le \sum_{n\le T}\Pr\left[S_{n,i}\cap Z_n \neq \emptyset\right] \\
&\le \sum_{n\le T: i_{\max}(n)\ge i} |Z_n|\cdot |S_{n,i}|/|S_{n,i-1}| \\
&\le O(T)\cdot \sum_{n: t^{-1}(T)\le n\le T} p(n) \\
&\le O(T^2\cdot t(t^{-1}(T))^{-6}) \\
&= O(T^{-4}).
\end{aligned}$$

Thus, we conclude that

$$\begin{aligned}
\Pr_{\mathcal{O}}\left[\mathcal{A}(M,x,1^{T^{2^c}}) \neq \mathcal{A}^*(M,x,1^{T^{2^c}})\right] &= \mathop{\mathbb{E}}_{C_T}\left[\Pr_{S_{1,i},\dots,S_{T,i}}\left[\phi'_{C_T}|_{\rho_{1,i},\dots,\rho_{T,i}} \equiv * \middle| C_T, \phi'_{C_T}|_{\rho^*_{1,i},\dots,\rho^*_{T,i}} \equiv 1\right]\right] \\
&\le O(T^{-4}).
\end{aligned}$$

□

Furthermore, we can show the average-case easiness of NP under the oracle access to $\mathcal{A}^*$ (instead of $\mathcal{A}$). This part is almost same as the proof in Section 9.1 of the average-case easiness of PH in the errorless setting.

Lemma 9.2.13. *Let M be a $t_M(n)$-time nondeterministic oracle machine and S be a randomized polynomial-time oracle sampling machine. We assume that $S(1^n)$ takes at most $t_S(n)$ time to generate an instance of length n. Then, the following event occurs with probability 1 over the choice of $\mathcal{O}$: for any $n \in \mathbb{N}$,*

$$\Pr_{x\leftarrow S^{\mathcal{O}}(1^n)}\left[M^{\mathcal{O}}(x) \neq \mathcal{A}^*(M,x,1^{T^{2^c}})\right] \le O(n^{-4}),$$

where $T = \max\{n^{2^c}, t_M(n)^{2^c}, t_S(n)^{2^c}\}$.

Proof. Fix $n \in \mathbb{N}$ arbitrarily. Let $T = \max\{n^{2^c}, t_M(n)^{2^c}, t_S(n)^{2^c}\}$ and $i_T = c^{-1}\log\log T$.

We first show that the instance $x \in \{0,1\}^n$ generated by $S^{\mathcal{O}}(1^n)$ is determined by only $\rho_{n',j}$ for $n' \le T$ and $j \le i_T - 1$ with probability at least $1 - O(n^{-5})$. Then, we show that $\mathcal{A}^*(M,x,1^{T^{2^c}})$ returns $M^{\mathcal{O}}(x)$ with probability at least $1 - O(n^{-5})$ under the condition that x is determined by

only $\rho_{n',j}$ for $n' \leq T$ and $j \leq i_T - 1$. If we assume these, then by the union bound, we have the lemma as

$$\Pr_{\mathcal{O},x \leftarrow S^{\mathcal{O}}(1^n)}\left[M^{\mathcal{O}}(x) \neq \mathcal{A}^*(M,x,1^{T^{2^c}})\right] \leq O(n^{-5}) + O(n^{-5}) = O(n^{-5})\ (= O(n^{-4})).$$

Now, we show the first claim. Since $t_S(n) \leq T^{1/2^c}$, the answers of $\mathcal{A}$ to queries made by $S^{\mathcal{O}}(1^n)$ are determined by only $\rho_{n',j}$ for $n' \leq T^{1/2^c}$ and $j \leq i_T - 2$. Under an arbitrary condition on restrictions $\rho_{n',j}$ for $n' \leq T^{1/2^c}$ and $j \leq i_{t_S(n)^{1/2^c}} \leq i_T - 2$, the output $S^{\mathcal{O}}(1^n)$ is determined by only $\rho_{n',j}$ for $n' \leq T$ and $j \leq i_T - 1$ unless S queries $(z,x,\ell) \in \cup_{n' \leq T^{1/2^c}} {\rho^*_{n',i_T-1}}^{-1}(*)$ to $\mathcal{F}$. Note that, if $n' \in \mathbb{N}$ satisfies $n' < t^{-1}(T^{1/2^c})$, then we have $i_{\max}(n') < c^{-1}\log\log(T^{1/2^c}) = c^{-1}\log\log T - 1 = i_T - 1$. Thus, ${\rho^*_{n',i_T-1}}^{-1}(*) = \emptyset$. Otherwise, each element in ${\rho^*_{n',i_T-1}}^{-1}(*)$ is selected from V_{n',i_T-2} independently with probability $p(n')$. Since $S^{\mathcal{O}}(1^n)$ accesses to $\mathcal{A}$ at most $T^{1/2^c}$ times, such a conditional probability is bounded above by

$$\begin{aligned} T^{1/2^c} \max_{t^{-1}(T^{1/2^c}) \leq n' \leq T^{1/2^c}} p(n') &= T^{1/2^c} p(t^{-1}(T^{1/2^c})) \\ &= T^{1/2^c} t(t^{-1}(T^{1/2^c}))^{-6} \\ &= (T^{1/2^c})^{-5} \\ &\leq n^{-5}, \end{aligned}$$

where the inequality holds because $T \geq n^{2^c}$.

Next, we show the second claim. Under the condition that the given instance $x \in \{0,1\}^n$ is determined by only $\rho_{n',j}$ for $n' \leq T$ and $j \leq i_T - 1$, the T-DNF formula ϕ constructed in $\mathcal{A}^*(M,x,1^{T^{2^c}})$ is determined only by $\rho_{n',j}$ for $n' \leq T$ and $j \leq i_T - 1$. Then, applying the restriction $\rho_{n',j}$ for $n' \leq T$ and $j \leq i_T$ under this condition is regarded as a $p(n')$-random restriction to V_{n',i_T-1} for each $n' \leq T$, where we can ignore small n' such that $n' < t^{-1}(T)$ because $i_{\max}(n') < c^{-1}\log\log T = i_T$ for such n'. By applying the switching lemma (particularly, the baby switching lemma) for T-DNF [Hås86], the probability that ϕ does not become a constant is at most

$$\begin{aligned} O\left(T \max_{t^{-1}(T) \leq n' \leq T} p(n')\right) &= O\left(T \cdot t(t^{-1}(T))^{-6}\right) \\ &= O(T^{-5}) \\ &= O(n^{-5}), \end{aligned}$$

where the last equation holds because $T \geq n^{2^c} \geq n$. Remember that $\mathcal{A}^*$ always returns the correct answer whenever ϕ becomes constant. Therefore, the second claim holds. □

Now, we derive the average-case easiness of NP from Lemmas 9.2.12 and 9.2.13 .

Proof of Theorem 9.2.11. We consider an arbitrary distributional NP problem $(L,\mathcal{D})$ and assume that L is specified by a $t_M(n)$-time nondeterministic oracle machine M, and $\mathcal{D}$ is specified by a randomized $t_S(n)$-time oracle sampling machine S. Let $T = \max\{n^{2^c}, t_M(n)^{2^c}, t_S(n)^{2^c}\}$ as in Lemma 9.2.13.

We construct an error-prone heuristic algorithm B for $(L,\mathcal{D})$ as follows: On input $x \in \{0,1\}^n$, B queries $(M,x,1^{T^{2^c}})$ to $\mathcal{A}$ and returns the same answer. In the following, we will verify that the

error probability of B (over the choice of $\mathcal{O}$ and $S^{\mathcal{O}}(1^n)$) is bounded above by $O(n^{-4})$ for each input size n. Then, by applying Markov's inequality and the Borel–Cantelli lemma, the error probability of B is bounded above by n^{-2} for all sufficiently large n with probability 1 over the choice of $\mathcal{O}$. Since the number of tuples (M, S) is countable, we can conclude that all distributional NP problems have an error-prone heuristic algorithm with error probability at most n^{-2} with probability 1 over the choice of $\mathcal{O}$. Based on the argument in [Imp95, Proposition 3], this is sufficient for the statement that $\mathsf{DistNP}^{\mathcal{O}} \subseteq \mathsf{HeurP}^{\mathcal{O}}$.

Therefore, it is sufficient to show that, for any $n \in \mathbb{N}$,

$$\Pr_{\mathcal{O},x\leftarrow S^{\mathcal{O}}(1^n)}\left[M^{\mathcal{O}}(x) \neq \mathcal{A}(M, x, 1^{T^{2^c}})\right] \leq O(n^{-4}).$$

Obviously, this event occurs only if (i)$\mathcal{A}(M, x, 1^{T^{2^c}}) \neq \mathcal{A}^*(M, x, 1^{T^{2^c}})$ or (ii) $M^{\mathcal{O}}(x) \neq \mathcal{A}^*(M, x, 1^{T^{2^c}})$ occur. By Lemmas 9.2.12 and 9.2.13 and the union bound, we have

$$\begin{aligned}
&\Pr_{\mathcal{O},x\leftarrow S^{\mathcal{O}}(1^n)}\left[M^{\mathcal{O}}(x) \neq \mathcal{A}(M, x, 1^{T^{2^c}})\right] \\
&\quad\leq \Pr_{\mathcal{O},x\leftarrow S^{\mathcal{O}}(1^n)}\left[\mathcal{A}(M, x, 1^{T^{2^c}}) \ \neq \mathcal{A}^*(M, x, 1^{T^{2^c}})\right] + \Pr_{\mathcal{O},x\leftarrow S^{\mathcal{O}}(1^n)}\left[M^{\mathcal{O}}(x) \neq \mathcal{A}^*(M, x, 1^{T^{2^c}})\right] \\
&\quad= O(T^{-4}) + O(n^{-4}) = O(n^{-4}) + O(n^{-4}) = O(n^{-4}).
\end{aligned}$$

□

9.2.5 Errorless Average-Case Hardness of NP

In this section, we show the hardness part of our oracle separation. First, we show the existence of AIPRG relative to $\mathcal{O}_a$. Then we show the other hardness results, including the errorless average-case hardness for NP, as corollaries.

We use the following theorem, which shows the existence of PRGs based on a random oracle.

Theorem 9.2.14 ([Zim98])**.** *For each $n \in \mathbb{N}$, let $\mathcal{R}_n \colon \{0,1\}^n \to \{0,1\}^n$ be a random function oracle, i.e., $\mathcal{R}_n$ is selected uniformly at random from $\{f \colon \{0,1\}^n \to \{0,1\}^n\}$.*

There exist a polynomial-time deterministic oracle machine $G^?$ and constants $c \geq 1$ and $b, \epsilon > 0$ satisfying the following: For any $n \in \mathbb{N}$ and $x \in \{0,1\}^{cn}$, $G^{\mathcal{R}_n}(x)$ generates a binary string of length $4cn$, and all oracle circuits $C^?$ of size 2^{bn} satisfy that

$$\left|\Pr_{U_{cn}}\left[C^{\mathcal{R}_n}(G^{\mathcal{R}_n}(U_{cn})) = 1\right] - \Pr_{U_{4cn}}\left[C^{\mathcal{R}_n}(U_{4cn}) = 1\right]\right| \leq 2^{-\epsilon n},$$

with probability at least $1 - 2^n$ over the choice of $\mathcal{R}_n$.

Furthermore, the result above is relativized, i.e., the above holds in the presence of an arbitrary oracle $\mathcal{O}$ independent of the choice of R_n.

Now, we show the existence of AIPRG relative to $\mathcal{O}_a$.

Theorem 9.2.15. *Let $a > 0$ be an arbitrary constant. With probability 1 over the choice of $\mathcal{O}_a$, there exists an AIPRG $G^{\mathcal{O}_a} = \{G_z^{\mathcal{O}_a}\}_{z\in\{0,1\}^*}$ against $\mathsf{SIZE}^{\mathcal{O}_a}[2^{\epsilon an/\log n}]$ for some absolute constant $\epsilon > 0$, where $G_z^{\mathcal{O}_a} \colon \{0,1\}^{|z|} \to \{0,1\}^{3|z|}$ for each $z \in \{0,1\}^*$.*

Proof. Let $G^?$ and c be the oracle machine and the constant in Theorem 9.2.14, respectively. Then, we define an AIPRG $G' = \{G'_z\}_{z\in\{0,1\}^*}$ as $G'^{\mathcal{O}}_z(x) = G^{\mathcal{F}_{z'}}(x')_{\le 3|z|}$, where $x \in \{0,1\}^{|z|}$, $z' = z_{\le\lfloor |z|/c\rfloor}$, $x' = x_{\le c|z'|}$, and $\mathcal{F}_{z'} \colon \{0,1\}^{|z'|} \to \{0,1\}^{|z'|}$ is defined as $\mathcal{F}_{z'}(y) = \mathcal{F}(z', y, 1) \circ \cdots \circ \mathcal{F}(z', y, |z'|)$. The validity of the truncation is verified as that $|x'| = c|z'| \le c \cdot |z|/c = |z| = |x|$ and $|G^{\mathcal{F}_{z'}}(x')| = 4|x'| = 4c|z'| \ge 4c(|z|/c - 1) \ge 4|z| - 4c \ge 3|z|$ for any z with $|z| \ge 4c$.

Let $\epsilon = 1/2c$ and $s(n) = 2^{\epsilon an/\log n}$. We show that G' above is an AIPRG against $\mathsf{SIZE}^{\mathcal{O}}[s(n)]$. Suppose there exists a family $C = \{C_n\}_{n\in\mathbb{N}}$ of oracle circuits of size $s(n)$ that breaks G', i.e., for any sufficiently large $n \in \mathbb{N}$ and any $z \in \{0,1\}^n$,

$$\left|\Pr_{U_n}\left[C^{\mathcal{O}}_n(z, G'^{\mathcal{O}}_z(U_n)) = 1\right] - \Pr_{U_{3n}}\left[C^{\mathcal{O}}_n(z, U_{3n}) = 1\right]\right| > \frac{1}{\mathsf{poly}(n)}.$$

Fix $n \in \mathbb{N}$ arbitrarily, and let $n' = \lfloor n/c\rfloor$. We consider an arbitrary choice of $\mathcal{O}$ except for the values of $\rho_{n',i_{\max}(n')}$ (we write this condition as R for convenience). Fix $z \in \{0,1\}^n$ such that $z' = z_{\le n'} \in S_{n',i_{\max}(n')-1}$ arbitrary (where $S_{\cdot,\cdot}$ is the random set in the oracle construction). We refer to such z as a hard index.

Since the size of C_n is at most $s(n) = 2^{\frac{an}{2c\log n}} \le 2^{\frac{a(\lfloor n/c\rfloor)}{\log(\lfloor n/c\rfloor)}} = t(n')$ for sufficiently large n (where t is the time-bound function in the oracle construction), the answers to queries by C_n of the form $\mathcal{A}(M, y, 1^{T^{2^c}})$ do not depend on the values of $\mathcal{F}_{z'}$ and are determined by condition R because they are determined only by the restrictions $\rho_{\cdot,j}$ for $j \le c^{-1}\log\log T \le c^{-1}\log\log s(n)^{1/2^c} \le c\log\log t(n') - 1 < i_{\max}(n')$. Now, we consider an arbitrary choice of $\rho_{n',i_{\max}(n')}$ except for the values of $\mathcal{F}_{z'}$ and denote this condition by R'. It is not hard to verify that $\mathcal{F}_{z'}$ is selected uniformly at random even under the conditions R and R'. Therefore, under the conditions R and R', we can identify the query access to $\mathcal{O}$ by C with the query access to another oracle $\mathcal{O}'$ (determined only by R and R') and a random function oracle $\mathcal{F}_{z'}$ that are selected independently of $\mathcal{O}'$.

For any $n \in \mathbb{N}$, let E_n be an event (over the choice of $\mathcal{F}_{z'}$) that there exists a circuit C' of size $2^{bn'}$, where b represents the constant in Theorem 9.2.14, such that

$$\begin{aligned}&\left|\Pr_{U_n}\left[C'^{\mathcal{O}}(G'^{\mathcal{F}_{z'}}_z(U_n)) = 1\right] - \Pr_{U_{3n}}\left[C'^{\mathcal{O}}(U_{3n}) = 1\right]\right| \\ &\qquad = \left|\Pr_{U_n}\left[C'^{\mathcal{O}',\mathcal{F}_{z'}}(G^{\mathcal{F}_{z'}}(U_n)) = 1\right] - \Pr_{U_{3n}}\left[C'^{\mathcal{O}',\mathcal{F}_{z'}}(U_{3n}) = 1\right]\right| > \frac{1}{\mathsf{poly}(n)}.\end{aligned}$$

Then, by Theorem 9.2.14 (relative to $\mathcal{O}'$), we have $\Pr_{\mathcal{O}}[E_n | R, R'] \le 2^{-\Omega(n)}$. By the Borel–Cantelli lemma, E_n occurs only for finitely many n's with probability 1 over the choice of $\mathcal{O}$ conditioned on R, R' (i.e., the choice of $\mathcal{F}_{z'}$). By taking the expectation over R, R', we can show that, with probability 1 over the choice of $\mathcal{O}$, there is no family C' of $2^{bn'}$-size circuits satisfying that for any sufficiently large $n \in \mathbb{N}$, there exists a hard index $z \in \{0,1\}^n$ such that

$$\left|\Pr_{U_n}\left[C'^{\mathcal{O}}(G'^{\mathcal{F}_{z'}}_z(U_n)) = 1\right] - \Pr_{U_{3n}}\left[C'^{\mathcal{O}}(U_{3n}) = 1\right]\right| > \frac{1}{\mathsf{poly}(n)}.$$

However, the circuit C in the assumption violates this statement by embedding a hard index z as auxiliary-input because the size is at most $n + s(n) = O(2^{(a/2c)\cdot n/\log n}) = o(2^{bn'})$. Therefore, we conclude that, with probability 1 over the choice of $\mathcal{O}$, there is no such a circuit C of size $s(n)$, and G' is an AIPRG against $\mathsf{SIZE}[s(n)]$. □

Now, we present the consequences of the existence of AIPRG. First, it is the well-established that any PRG is also OWF [cf. Gol01, Proposition 3.3.8]. This result is trivially extended to the case of auxiliary-input primitives, and the following holds.

Corollary 9.2.16. *Let $a > 0$ be an arbitrary constant. With probability 1 over the choice of $\mathcal{O}_a$, there exists an AIOWF against $\mathsf{SIZE}^{\mathcal{O}_a}[2^{\epsilon an/\log n}]$ relative to $\mathcal{O}_a$ for some absolute constant $\epsilon > 0$.*

Since AIPRG implies HSG by regarding the auxiliary-input as a part of the hidden input to HSG, we also obtain the existence of HSG.

Corollary 9.2.17. *Let $a > 0$ be an arbitrary constant. With probability 1 over the choice of $\mathcal{O}_a$, there exists an HSG $G^{\mathcal{O}_a} = \{G_n^{\mathcal{O}_a}\}_{n\in\mathbb{N}}$ against $\mathsf{SIZE}^{\mathcal{O}_a}[2^{\epsilon an/\log n}]$ for some absolute constant $\epsilon > 0$, where $G_n^{\mathcal{O}_a} \colon \{0,1\}^{2n} \to \{0,1\}^{3n}$ for each $n \in \mathbb{N}$.*

Proof. It is easily observed that the proof of Lemma 8.4.1 is relativizing. The corollary immediately follows from Corollary 9.2.16 and the relativized version of Lemma 8.4.1. □

Furthermore, the existence of HSGs implies the errorless average-case hardness of NP, as observed in [HS17] and the proof of Lemma 8.4.2.

Corollary 9.2.18. *Let $a > 0$ be an arbitrary constant. With probability 1 over the choice of $\mathcal{O}_a$, $\mathsf{DistNP}^{\mathcal{O}_a} \not\subseteq \mathsf{AvgSIZE}^{\mathcal{O}_a}[2^{\frac{\epsilon an}{\log n}}]$ for some absolute constant $\epsilon > 0$.*

Proof. Let $G^{\mathcal{O}}$ and ϵ be the HSG and the constant in Corollary 9.2.17, respectively. Then, we define the language $L^{\mathcal{O}}$ as $L^{\mathcal{O}} := \mathrm{Im}G^{\mathcal{O}}$. Obviously, $L^{\mathcal{O}} \in \mathsf{NP}^{\mathcal{O}}$ and $(L^{\mathcal{O}}, \{U_n\}_{n\in\mathbb{N}}) \in \mathsf{DistNP}^{\mathcal{O}}$. Thus, it is sufficient to show that $(L^{\mathcal{O}}, \{U_n\}_{n\in\mathbb{N}}) \notin \mathsf{Avg}_{1/4}\mathsf{SIZE}^{\mathcal{O}}[2^{\frac{\epsilon an}{4\log n}}]$.

For contradiction, we assume that $(L^{\mathcal{O}}, \{U_n\}_{n\in\mathbb{N}}) \in \mathsf{Avg}_{1/4}\mathsf{SIZE}^{\mathcal{O}}[2^{\frac{\epsilon an}{4\log n}}]$. Then, there exists a family $C = \{C_n\}_{n\in\mathbb{N}}$ of $O(2^{\frac{\epsilon an}{4\log n}})$-size oracle circuits for $(L^{\mathcal{O}}, \{U_n\}_{n\in\mathbb{N}})$, i.e., for any sufficiently large $n \in \mathbb{N}$,

$$\Pr_{y\sim\{0,1\}^{3n}}\left[C_{3n}^{\mathcal{O}}(y) = \bot\right] \le 1/4$$

and for each $y \in \{0,1\}^{3n}$,

$$C_{3n}^{\mathcal{O}}(y) \in \{\mathbb{1}(y \in L^{\mathcal{O}}), \bot\}.$$

Note that the size of C_{3n} is at most $O(2^{\frac{\epsilon a\cdot 3n}{4\log 3n}})$.

Next, we define an adversary C' for $G^{\mathcal{O}}$ as follows: for a given $y \in \{0,1\}^{3n}$ (i.e., the length of seed is $2n$), $C_{2n}'^{\mathcal{O}}$ simulates $C_{3n}^{\mathcal{O}}(y)$, and if C_{3n} returns 1 or $\bot$, then C'_{2n} outputs 0; otherwise (i.e., if C_{3n} returns 0), C'_{2n} outputs 1. Then, based on the aforementioned inequalities, it is not hard to verify that for any sufficiently large $n \in \mathbb{N}$,

$$\Pr_{y\sim\{0,1\}^{3n}}\left[C_{2n}'^{\mathcal{O}}(y) = 0\right] \le \frac{1}{4} + \frac{|\{G^{\mathcal{O}}(x) : x \in \{0,1\}^{2n}\}|}{|\{0,1\}^{3n}|} \le \frac{1}{4} + \frac{2^{2n}}{2^{3n}} < \frac{1}{2},$$

and for any $x \in \{0,1\}^{2n}$, we have $C_{3n}^{\mathcal{O}}(G^{\mathcal{O}}(x)) \in \{1, \bot\}$ and $C_{2n}'^{\mathcal{O}}(G^{\mathcal{O}}(x)) = 0$.

Therefore, C' succeeds in avoiding $\mathrm{Im}G^{\mathcal{O}}$, and the size is bounded above by $O(2^{\frac{\epsilon an}{\log n}}) = O(2^{\frac{\epsilon a(2n)}{\log(2n)}})$. This contradicts Corollary 9.2.17. Thus, we conclude that $(L^{\mathcal{O}}, \{U_n\}_{n\in\mathbb{N}}) \notin \mathsf{Avg}_{1/4}\mathsf{SIZE}^{\mathcal{O}}[2^{\frac{\epsilon an}{4\log n}}]$. □

Furthermore, based on the GGM construction [GGM86], we can translate AIPRGs into AIPRFs. In the security proof, the seed length is preserved, and an adversary of size $s(n)$ for the PRF is translated into an adversary of size $s(n) \cdot \mathsf{poly}(n)$ for the original PRG, where poly is a polynomial depending on the computational cost of the PRG. This observation implies the following:

Corollary 9.2.19. *Let $a > 0$ be an arbitrary constant. With probability 1 over the choice of $\mathcal{O}_a$, there exists an AIPRF $f^{\mathcal{O}_a} = \{f_z^{\mathcal{O}_a}\}_{z\in\{0,1\}^*}$ against $\mathsf{SIZE}^{\mathcal{O}_a}[2^{\epsilon an/\log n}]$ for some absolute constant $\epsilon > 0$, where $f_z^{\mathcal{O}_a} : \{0,1\}^{|z|} \times \{0,1\}^{|z|} \to \{0,1\}$ for each $z \in \{0,1\}^*$.*

In Section 5.2, we proved that the existence of AIPRF implies the average-case hardness of distribution-free learning, where the complexity of the concept class depends on the complexity of computing AIPRF. Thus, we obtain the following, where we apply the standard transformation from nonuniform Turing machines to circuit families.

Corollary 9.2.20. *There exist a polynomial $s(n)$ and a constant $\epsilon > 0$ such that for any $a > 0$, $\mathsf{SIZE}[s(n)]$ is not PAC learnable on average in the BFKL model with respect to all unknown example distributions and a fixed samplable distribution over $\mathsf{SIZE}[s(n)]$ by nonuniform $O(2^{\epsilon an/\log n})$-time algorithms with probability 1 over the choice of $\mathcal{O}_a$.*

Proof. It is easily observed that the proof of Theorem 5.2.1 is relativizing. The corollary immediately follows from Corollary 9.2.19 and the relativized version of Theorem 5.2.1. □

Without loss of generality, we can let $s(n) = n^b$ in above for some $b > 0$. By the simple padding argument, where we stretch an n-bit example into an $s(n)$-bit example, the size complexity of the target function becomes $O(n)$ (for the input length $s(n)$) in above. Since $2^{s(n)^{1/(b+1)}} = o(2^{\epsilon an/\log n})$, we have the following:

Corollary 9.2.21. *There exists $\epsilon > 0$ such that for any $a > 0$, $\mathsf{SIZE}[n]$ is not PAC learnable on average in the BFKL model with respect to all unknown example distribution and a fixed samplable distribution over $\mathsf{SIZE}[n]$ by nonuniform $O(2^{n^\epsilon})$-time algorithms with probability 1 over the choice of $\mathcal{O}_a$.*

The existence of AIPRF also implies the (worst-case) hardness of PAC learning $\mathsf{SIZE}[s(n)]$ on the uniform distribution, as observed in [ABX08], where $s(n)$ is a polynomial depending on the complexity of computing the AIPRF. In our case, we can directly construct a hard-to-learn class and show the hardness of learning $\mathsf{SIZE}[n]$.

Theorem 9.2.22. *Let $a > 0$ be an arbitrary constant. With probability 1 over the choice of $\mathcal{O}_a$, $\mathsf{SIZE}^{\mathcal{O}_a}[n]$ is not weakly PAC learnable with MQ on the uniform distribution by nonuniform $O(2^{\epsilon an/\log n})$-time algorithms relative to $\mathcal{O}^a$ for some absolute constant $\epsilon > 0$.*

The proof of Theorem 9.2.22 is an analog of the proof of Theorems 9.1.12 and 9.1.20. For completeness, we present the formal proof in Section 9.2.6.

Furthermore, Oliveira and Santhanam [OS17] showed the speedup phenomena in PAC learning with MQ on the uniform distribution. One of their results is stated below.

Theorem 9.2.23 (speedup lemma [OS17])**.** *For any polynomial $s(n)$ and constant $\epsilon > 0$, there exists a polynomial $s'(n)$ such that if $\mathsf{SIZE}[s(n)]$ is not weakly PAC learnable with MQ on uniform distribution by nonuniform $O(2^{n^\epsilon})$-time algorithms, then $\mathsf{SIZE}[s'(n)]$ is not weakly PAC learnable with MQ on uniform distribution by nonuniform $2^n/n^{\omega(1)}$-time algorithms. Furthermore, this result is relativized.*

Theorems 9.2.22 and 9.2.23 immediately imply the following.

Corollary 9.2.24. *There exists a polynomial $s(n)$ such that for any $a > 0$, $\mathsf{SIZE}[s(n)]$ is not PAC learnable with MQ on the uniform distribution by nonuniform $2^n/n^{\omega(1)}$-time algorithms with probability 1 over the choice of $\mathcal{O}_a$.*

Finally, we mention the hardness of approximation problems in meta-complexity. The hardness of GapMINKT follows from the existence of HSG in the same manner as Corollary 9.2.18.

Corollary 9.2.25. *Let $a > 0$ be an arbitrary constant. With probability 1 over the choice of $\mathcal{O}_a$, $\mathrm{Gap}_\sigma\mathrm{MINKT}^{\mathcal{O}_a} \notin \text{pr-}\mathsf{SIZE}^{\mathcal{O}_a}[2^{\epsilon an/\log n}]$ for any $\sigma(s,n) = o(s) \cdot \mathsf{polylog}(n)$, where $\epsilon > 0$ is an absolute constant.*

Proof sketch. Suppose that $\mathrm{Gap}_\sigma\mathrm{MINKT}^{\mathcal{O}} \in \text{pr-}\mathsf{SIZE}^{\mathcal{O}}[2^{(\epsilon/4)an/\log n}]$ for some $\sigma(s,n) = o(s) \cdot \mathsf{polylog}(n)$, where $\epsilon > 0$ is the constant in Corollary 9.2.17. Then, there exists an $O(2^{(\epsilon/4)an/\log n})$-size oracle circuit C for $\mathrm{Gap}_\sigma\mathrm{MINKT}^{\mathcal{O}}$. Based on C, we can construct an adversary for an arbitrary HSG $G^{\mathcal{O}}\colon \{0,1\}^{2n} \to \{0,1\}^{3n}$ because, for each $n \in \mathbb{N}$ and $x \in \{0,1\}^{2n}$, it holds that $\mathrm{K}^{t,\mathcal{O}}(G^{\mathcal{O}}(x)) \leq 2n + O(1)$ for a proper choice of $t = \mathsf{poly}(n)$ and $\Pr_{y\sim\{0,1\}^{3n}}[\mathrm{K}^{\mathcal{O}}(y) \geq 3n - 2\ (> 2n + O(1) + \sigma(2n + O(1), n))] \geq 3/4$. It is not hard to verify that the size of the adversary based on C is at most $O(2^{(\epsilon/4)3an/\log 3n})$. Thus, this contradicts Corollary 9.2.17. □

Furthermore, [CIKK16] constructed a PAC learning algorithm for $\mathsf{P/poly}$ with MQ on the uniform distribution based on an algorithm for GapMCSP (originally, the existence of natural proofs). As the contraposition, we obtain the following from Corollary 9.2.24.

Corollary 9.2.26. *Let $a > 0$ be an arbitrary constant. With probability 1 over the choice of $\mathcal{O}_a$, for each $\epsilon > 0$, there exists $\delta > 0$ such that $\mathrm{Gap}_\epsilon\mathrm{MCSP}^{\mathcal{O}_a} \notin \text{pr-}\mathsf{SIZE}^{\mathcal{O}_a}[2^{n^\delta}]$.*

Theorem 9.2.8 follows from Theorems 9.2.11, 9.2.15, and 9.2.22 and Corollaries 9.2.16– 9.2.26 by selecting an appropriately large parameter in the oracle construction according to $a > 0$ in the statement of Theorem 9.2.8.

9.2.6 Proof of Theorem 9.2.22

We present the formal proof of Theorem 9.2.22.

Proof of Theorem 9.2.22. For every choice of $\mathcal{O}_a$, we define a concept class $\mathscr{C}^{\mathcal{O}_a}$ as

$$\mathscr{C}^{\mathcal{O}_a} = \{\mathcal{F}_{|z|}(z,\cdot,1) : z \in \{0,1\}^*\}.$$

Then, we show that $\mathscr{C}^{\mathcal{O}_a}$ is not weakly PAC learnable with MQ on the uniform distribution by nonuniform $t_L(n) = O(2^{(a/2)n/\log n})$-time algorithms with probability 1 over the choice of $\mathcal{O}_a$. Since $\mathscr{C}^{\mathcal{O}_a} \subseteq \mathsf{SIZE}^{\mathcal{O}_a}[n]$, the theorem also holds.

Let $\epsilon(n) = n^{-\log n}$. We fix $n \in \mathbb{N}$ arbitrarily. We consider an arbitrary nonuniform randomized oracle machine (i.e., a learner) L. For each $z \in \{0,1\}^n$, we define I_z as an event (over the choice of $\mathcal{O}$) that L succeeds in learning $f_z(x) \equiv F_n(z,x,1) \in \mathscr{C}_n^{\mathcal{O}}$ in $t_L(n)$ time with advantage $\epsilon(n)$, i.e.,

$$I_z = \left(\Pr_L\left[L^{\mathcal{O},\mathsf{MQ}_{f_z}}(n) \to h^{\mathcal{O}} \text{ s.t. } \Pr_x[h^{\mathcal{O}}(x) = f_z(x)] \geq 1/2 + \epsilon(n) \text{ in } t_L(n) \text{ time}\right] \geq 2/3\right).$$

We will show that $\Pr_{\mathcal{O}}[\wedge_{z\in\{0,1\}^n} I_z] \le 2^{-2^{\Omega(n)}}$. For now, we assume this and show the hardness of learning $\mathscr{C}^{\mathcal{O}}$. Since any nonuniform $t_L(n)$-time oracle machine has a binary representation of length at most $O(t_L(n))$ (for each $n \in \mathbb{N}$), the event E_n that there exists a nonuniform $t_L(n)$-time oracle machine succeeds in learning $\mathscr{C}_n^{\mathcal{O}}$ is at most $2^{O(t_L(n))} \cdot 2^{-2^{\Omega(n)}} = \mathsf{negl}(n)$ by the union bound. By the Borel–Cantelli lemma, these events E_n occur only for finitely many $n \in \mathbb{N}$ with probability 1 over the choice of $\mathcal{O}$. In such cases, there is no nonuniform $t_L(n)$-time algorithm that succeeds in weak learning for $\mathscr{C}^{\mathcal{O}}$.

Now, we show that $\Pr_{\mathcal{O}}[\wedge_{z\in\{0,1\}^n} I_z] \le 2^{-2^{\Omega(n)}}$. For any $z, x \in \{0,1\}^n$, we use a notation $L^{\mathcal{O}}(n)(x)$ to refer to the following procedure: We execute $L^{\mathcal{O},\mathsf{MQ}_{f_z}}(n)$ and if L outputs some hypothesis $h^?$ in $t_L(n)$ time, then we also execute $h^{\mathcal{O}}(x)$. For any $z \in \{0,1\}^n$, we define an event J_z as the event (over the choice of $\mathcal{O}$) that L or its hypothesis directly access a target function f_z by $\mathcal{F}$, i.e.,

$$J_z = \left(\Pr_{L,x\sim\{0,1\}^n} \left[\mathcal{F}(z, x', \ell) \text{ is queried for some } (x', \ell) \in \{0,1\}^n \times [n] \text{ during } L^{\mathcal{O}}(n)(x) \right] \ge \epsilon(n)^4 \right).$$

We say that $z \in \{0,1\}^n$ is a hard index (relative to $\mathcal{O}$) if $z \in S_{n,i_{\max}(n)-1}$. Then, we have that

$$\begin{aligned}
\Pr_{\mathcal{O}}\left[\bigwedge_{z\in\{0,1\}^n} I_z\right] &\le \Pr_{\mathcal{O}}\left[\bigwedge_{z\in\{0,1\}^n:\text{hard}} I_z\right] \\
&\le \Pr_{\mathcal{O}}\left[\bigwedge_{z\in\{0,1\}^n:\text{hard}} (I_z \vee J_z)\right] \\
&\le \Pr_{\mathcal{O}}\left[\bigwedge_{z:\text{hard}} J_z\right] + \Pr_{\mathcal{O}}\left[\exists z : \text{hard s.t. } I_z \wedge \neg J_z\right].
\end{aligned}$$

In the following, we show that each term is bounded above by $2^{-2^{\Omega(n)}}$, which implies the theorem.

Claim 9.2.27. $\Pr_{\mathcal{O}}\left[\bigwedge_{z:\mathit{hard}} J_z\right] \le 2^{-2^{\Omega(n)}}$.

Proof. Let $N = |S_{n,i_{\max}(n)-1}|$. For any choice of $S_{n,i_{\max}(n)-1}$, we can divide a random selection of $\rho_{n,i_{\max}(n)}$ into the following two steps without loss of generality: (i) select N random functions $x_1, \ldots, x_N \in \{0,1\}^{n2^n}$ uniformly at random (where we regard each x_j as a truth table of a mapping from n bits to n bits), and (ii) select a random bijection $b\colon S_{n,i_{\max}(n)-1} \to \{x_1, \ldots, x_N\}$ to assign each value of $F(z,\cdot,\cdot)$ as $F(z,\cdot,\cdot) \equiv b(z)$ for each $z \in S_{n,i_{\max}(n)-1}$.

We consider an arbitrary choice of $\mathcal{O}$ except for the aforementioned bijection b and use the notation C to refer to such a partial choice of $\mathcal{O}$. We regard C as a condition on the choice of $\mathcal{O}$. We say that the partial choice C is bad if there are two distinct indices $j_1, j_2 \in [N]$ such that $x_{i_1} = x_{i_2}$. Since $x_1, \ldots, x_N$ is uniformly and independently selected from 2^{n2^n} elements, by the union bound, we obtain that

$$\Pr_{\mathcal{O}}[C \text{ is bad}] \le N^2 \cdot 2^{-n2^n} \le 2^{2n} \cdot 2^{-n2^n} = 2^{-\Omega(2^n)}.$$

Thus, we have that

$$\begin{aligned}\Pr_{\mathcal{O}}\left[\bigwedge_{z:\text{hard}} J_z\right] &= \mathop{\mathbb{E}}_{C}\left[\Pr_{\mathcal{O}}\left[\bigwedge_{z:\text{hard}} J_z \middle| C\right]\right] \\ &\le \mathop{\mathbb{E}}_{C}\left[\Pr_{\mathcal{O}}\left[\bigwedge_{z:\text{hard}} J_z \middle| C\right] \middle| C \text{ is not bad}\right] + \Pr_{O}\left[C \text{ is bad}\right] \\ &\le \mathop{\mathbb{E}}_{C}\left[\Pr_{b}\left[\bigwedge_{z:\text{hard}} J_z \middle| C\right] \middle| C \text{ is not bad}\right] + 2^{-\Omega(2^n)}.\end{aligned}$$

Therefore, it is sufficient to show that for every not bad condition C,

$$\Pr_{b}\left[\bigwedge_{z:\text{hard}} J_z \middle| C\right] = 2^{-2^{\Omega(n)}}.$$

To show the aforementioned bound, we assume that $\bigwedge_{z:\text{hard}} J_z$ holds under a not bad condition C (notice that C determines all hard indices), i.e., for any hard index $z \in \{0,1\}^n$,

$$\Pr_{L,x\sim\{0,1\}^n}\left[\mathcal{F}(z,\cdot,\cdot) \text{ is queried during } L^{\mathcal{O}}(n)(x)\right] \ge \epsilon(n)^4$$

By the standard probabilistic argument, we can reduce the upper bound from $1-\epsilon(n)^4$ to 2^{-2n} on the probability that $\mathcal{F}(z,\cdot,\cdot)$ is not queried by L or its hypothesis by repeating $L^{\mathcal{O}}(n)(x)$ $2n/\epsilon(n)^4$ times. Then, by the union bound for all hard indices, there exists a random seed $r \in \{0,1\}^{t_L(n)\cdot 2n/\epsilon(n)^4}$ such that for any hard index z, $\mathcal{F}(z,\cdot,\cdot)$ is queried during at least one execution of $L^{\mathcal{O}}(n)(x)$ by using the randomness r. We remark that all queries to $\mathcal{O}$ by L or its hypothesis are determined by C except for $\mathcal{F}(z,\cdot,\cdot)$ for each hard index z. This is because L and its hypothesis are executed in time $O(t_L(n)) = O(2^{(a/2)n/\log n}) \le 2^{an/\log n} = t(n)$ (for sufficiently large n), i.e., all answers from $\mathcal{A}$ depend on only $\rho_{\cdot,j}$ for $j \le c^{-1}\log\log t(n)^{1/2^c} = c^{-1}\log\log t(n) - 1 < i_{\max}(n)$.

Therefore, by executing L with the randomness r and tracing queries to $\mathcal{F}$, we can obtain a deterministic inverter for b of the query complexity at most $t_L(n)\cdot 2n/\epsilon(n)^4$, where the inverter simulates membership queries by using its input and its own query access to b. Particularly, the inverter accesses b only for answering the queries of the form $\mathcal{F}(z',\cdot,\cdot)$ for some $z' \in S_{n,i_{\max}(n)-1}$. However, by Lemma 9.1.11, such a bijection b is represented by $2\log\binom{N}{a} + \log((N-a)!)$ bits, where $a = N/(t_L(n)\cdot 2n\epsilon(n)^{-4}+1)$, when L and r are given. Thus, we obtain that

$$a \ge \frac{2^n\cdot p(n)^{i_{\max}(n)}}{O(t_L(n)n\epsilon(n)^{-4})} \ge \frac{2^n\cdot 2^{-\frac{6an}{\log n}\cdot\frac{1}{c}\log n}}{O(2^{(a/2)n/\log n} n^{4\log n+1})} \ge \frac{2^n\cdot 2^{-\frac{6}{7}n}}{2^{O(n/\log n)}} \ge 2^{\Omega(n)},$$

and

$$\begin{aligned}\Pr_{b}\left[\bigwedge_{z:\text{hard}} J_z \middle| C\right] &\le \frac{\binom{N}{a}^2\cdot(N-a)!\cdot 2^{O(t_L(n)\cdot 2n\epsilon(n)^{-4})}}{N!} \\ &\le \binom{N}{a}\cdot\frac{1}{a!}\cdot 2^{O(2^{(a/2)n/\log n})} \\ &\le \left(\frac{Ne}{a}\right)^a\cdot\frac{1}{\sqrt{2\pi a}}\left(\frac{e}{a}\right)^a\cdot 2^{2^{O(n/\log n)}}\end{aligned}$$

$$\leq \left(e(t_L(n)\cdot 2n\epsilon(n)^{-4}+1)\right)^a \cdot \left(\frac{e}{a}\right)^a \cdot 2^{2^{O(n/\log n)}}$$
$$\leq \left(\frac{2^{O(n/\log n)}}{a}\right)^a \cdot 2^{2^{O(n/\log n)}}$$
$$\leq 2^{-a}\cdot 2^{2^{O(n/\log n)}} = 2^{-2^{\Omega(n)}}.$$

$\diamond$

Claim 9.2.28. $\Pr_{\mathcal{O}}[\exists z : \textit{hard s.t. } I_z \wedge \neg J_z] \leq 2^{-2^{\Omega(n)}}$.

Proof. We consider an arbitrary choice of $\mathcal{O}$ except for values of $\rho_{n,i_{\max}(n)}$ (we write this condition as C). Note that hard indices are determined by the condition C, and for any hard index $z \in \{0,1\}^n$, f_z is a random function even under the condition C.

Suppose that z is a hard index, and $\neg I_z \wedge J_z$ occurs. By Markov's inequality, we derive the following from $\neg I_z$:

$$\Pr_L\left[\Pr_x\left[\mathcal{F}(z,\cdot,\cdot) \text{ is queried during } L^{\mathcal{O},\mathsf{MQ}_{f_z}}(n)(x)\right] \leq 4\epsilon(n)^3\right] \geq 1-\epsilon(n)/4.$$

Since J_z holds, we also have

$$\Pr_L\left[L^{\mathcal{O},\mathsf{MQ}_{f_z}}(n) \to h^{\mathcal{O}} \text{ s.t. } \Pr_x[h^{\mathcal{O}}(x)=f_z(x)] \geq 1/2+\epsilon(n) \text{ in } t_L(n) \text{ time }\right] \geq \frac{2}{3}.$$

From the aforementioned two inequalities, there exists a random string r for L such that

- $L^{\mathcal{O},\mathsf{MQ}_{f_z}}(n;r)$ outputs some hypothesis $h^{\mathcal{O}}$ in $t_L(n)$ time without querying $(z,\cdot,\cdot)$ to $\mathcal{F}$;
- $\Pr_x[h^{\mathcal{O}}(x) \text{ queries } (z,\cdot,\cdot) \text{ to } \mathcal{F}] \leq 4\epsilon(n)^3$; and
- $\Pr_x[h^{\mathcal{O}}(x)=f_z(x)] \geq 1/2+\epsilon(n)$.

Since L and h are only executed in $O(t_L(n)) \leq t(n)$ time (for sufficiently large n), any query $(M,x',1^{T^{2^c}})$ to $\mathcal{A}$ by L and h satisfies that $T^{2^c} \leq t(n)$ and $i_T = c^{-1}\log\log T = c^{-1}\log\log t(n) - 1 < i_{\max}(n)$. Therefore, if L and h do not query $(z,\cdot,\cdot)$ to $\mathcal{F}$, then the answers from $\mathcal{O}$ do not depend on $\rho_{n,i_{\max}(n)}$, i.e., they are determined only by the condition C.

In this case, we show that a truth table of f_z has a short description under the condition C. This implies the upper bound on the probability of this case because a random function does not have such a short description with extremely high probability.

The short description of f_z is obtained as follows. Let $B_z \subseteq \{0,1\}^n$ be the subset consisting of x such that $h^{\mathcal{O}}(x)$ queries $\mathcal{F}(z,\cdot,\cdot)$. By the second property of the above, $|B_z| \leq 2^n \cdot 4\epsilon(n)^3$ holds. We execute $L^{\mathcal{O},\mathsf{MQ}_{f_z}}(n;r)$ to obtain $h^{\mathcal{O}}$, and we write down all answers from the membership query oracle MQ_{f_z} in Q, i.e., Q is a binary string of length at most $t_L(n)$. By the first property on r, the answers from $\mathcal{O}$ are determined by the condition C. Next, we execute the outputted hypothesis $h^{\mathcal{O}}(x)$ on each input $x \in \{0,1\}^n \setminus B_z$. From these predictions and auxiliary information $f_z(B_z) = \{(x,f_z(x)) : x \in B_z\}$, we obtain a function $\tilde{f} : \{0,1\}^n \to \{0,1\}$ defined as

$$\tilde{f}(x) = \begin{cases} h^{\mathcal{O}}(x) & \text{if } x \notin B_z \\ f_z(x) & \text{if } x \in B_z. \end{cases}$$

Then, by the third property, $\tilde{f}$ is $(1/2-\epsilon(n))$-close to f_z. We define $e \in \{0,1\}^{2^n}$ as $e_{x+1} = f_z(x) \oplus \tilde{f}(x)$, where we identify $x \in \{0,1\}^n$ with an integer in $[0, 2^n-1]$. Then, the Hamming weight of e is at most $2^n \cdot (1/2-\epsilon(n))$, and e is represented by a binary string $\tilde{e}$ of length at most $(1-\Omega(\epsilon(n)^2))\cdot 2^n$ by lexicographic indexing among binary strings of the same weight. Obviously, f_z is reconstructed from $\tilde{f}$ and $\tilde{e}$. Therefore, based on the aforementioned constructions, f_z is represented only by $L, r, Q, f_z(B_z)$, and $\tilde{e}$ on the condition C. The total number of such representations is at most

$$\begin{aligned}|L| + t_L(n) + t_L(n) + (n+1)\cdot|B_z| + (1-\Omega(\epsilon(n)^2))\cdot 2^n &\le O(t_L(n)) + \left(1 + 4(n+1)\epsilon(n)^3 - \Omega(\epsilon(n)^2)\right) 2^n \\ &\le O(t_L(n)) + \left(1-\Omega(\epsilon(n)^2)\right)\cdot 2^n.\end{aligned}$$

Therefore, we have that for any condition C and any hard index $z \in \{0,1\}^n$,

$$\begin{aligned}\Pr_{\mathcal{O}}\left[I_z \wedge \neg J_z | C\right] &\le \frac{2^{O(t_L(n)) + \left(1-\Omega(\epsilon(n)^2)\right)\cdot 2^n}}{2^{2^n}} \\ &\le 2^{2^{O(n/\log n)} - \Omega(n^{-2\log n})\cdot 2^n} \\ &\le 2^{-2^{\Omega(n)}}.\end{aligned}$$

Thus, we conclude that

$$\begin{aligned}\Pr_{\mathcal{O}}\left[\bigvee_{z:\text{hard}} I_z \wedge \neg J_z\right] &= \mathop{\mathbb{E}}_{C}\left[\Pr_{\mathcal{O}}\left[\bigvee_{z:\text{hard}} I_z \wedge \neg J_z \middle| C\right]\right] \\ &\le \mathop{\mathbb{E}}_{C}\left[\sum_{z\in\{0,1\}^n} \Pr_{\mathcal{O}}\left[z \text{ is hard and } I_z \wedge \neg J_z | C\right]\right] \\ &\le 2^n \cdot 2^{-2^{\Omega(n)}} = 2^{-2^{\Omega(n)}}.\end{aligned}$$

◇

□

9.3 Related Work and Map of the Relativized World

The study of oracle separations is initiated by Baker, Gill, and Solovay [BGS75] to identify the barrier for resolving the P versus NP problem. The study of the average-case complexity is initiated by Levin [Lev86], and later it was brushed up by Impagliazzo [Imp95], where he introduced the notion of five worlds. In the same paper, Impagliazzo first addressed the question on the difference between the errorless complexity and the error-prone complexity. Each relativized world in Impagliazzo's five worlds is found in [BGS75; Imp11; Wee06; IR89; Bra83]. Specifically, Impagliazzo found a relativized heuristica in which DistNP $\subseteq$ AvgP but NP $\not\subseteq$ SIZE$[2^{n^\epsilon}]$ for some $\epsilon > 0$, and Wee found a relativized pessiland in which DistNP $\not\subseteq$ HeurP, but neither AIOWF nor OWF exists. Watson [Wat12] also constructed a relativized world in which there is no black-box worst-case to average-case reduction for NP, but the reduction presented by Hirahara [Hir18; Hir20b] is non-black-box and overcomes the barrier against black-box reductions. The work in Section 9.1 improved the oracle construction proposed by Impagliazzo to the tight worst-case hardness of NP

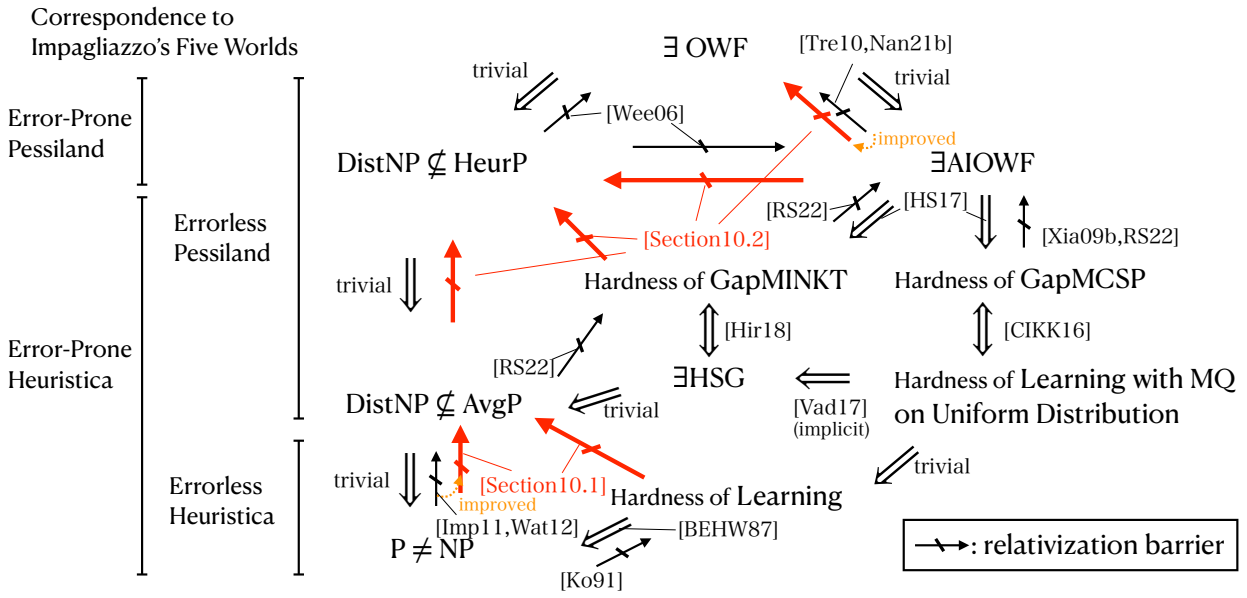

Figure 9.1: Relativization Barriers in Heuristica and Pessiland

and also presented the relativized world in which DistNP $\subseteq$ AvgP, but PAC learning P/poly with MQ is sub-exponentially hard. Ko [Ko91] showed the relativized world in which P $\neq$ NP, but a gap variant of the problem called MINLT is efficiently solvable, which is sufficient for PAC learning P/poly. Xiao [Xia09b] found the relativized world in which PAC learning P/poly with MQ is hard, but there is no AIOWF. Ren and Santhanam [RS22] presented various relativization barriers on the problems in meta-complexity, including the relativized world in which there is no efficient and robust reduction from distributional NP problems to the GapMINKT oracle. They also found the relativized world in which no AIOWF exists but GapMCSP and GapMINKT are hard (even in the error-prone average case). Trevisan [Tre10] and Nanashima [Nan21b] presented the oracle separation between AIOWF and OWF, which was conceptually improved by the result in Section 9.2. The relationships among these oracle separation results is visualized in Figure 9.1.

Chapter 10

Conclusions and Future Directions

In this thesis, we presented new connections between learning, average-case complexity, and cryptography, which provide new and clear insights into the gaps in our current knowledge between the computational hardness of NP and the existence of one-way functions. Based on the main results, we propose several future directions, including the learning-theoretic approach towards basing one-way functions on worst-case/average-case hardness of NP.

One of the most important open problems is, of course, to establish a non-relativized technique that breaks relativization barriers presented in Chapter 9. Additionally, it is also important to obtain tight connections in the relativized world to understand more deeply the capability of relativizing proof techniques. Although we identify the capabilities of relativizing techniques for establishing worst-case-to-average-case reductions for PH (in Section 9.1) and weak learning under the uniform example distribution in error-prone Heuristica (in Section 9.2), it remains unclear whether other separation results are tight or not. Below, we present specific open problems, which are classified into three realms, errorless Heuristica, error-prone Heuristica, and Pessiland.

Quest for Errorless Heuristica

An important open problem is to improve the size of hypotheses produced by a learner constructed under the average-case errorless easiness of NP. Our learning strategy presented in Section 3.1 implies a weak learner that learns all s-size programs by $O(sn)$-size programs under example distributions samplable with advice complexity $O(s)$ by using the standard hybrid argument instead of the characterization result in [KL18] or distribution-specific boosting in [Fel10] (see [Hir22a, Appendix A] for more formal arguments). Improving the size of hypotheses from $O(sn)$ to $s \cdot n^{(\log \log n)^{-c}}$ for a sufficiently large constant $c > 0$ implies excluding errorless Heuristica because of the NP-hardness result (i.e., Theorem 2.6.5) in [Hir22a]. Proving the implication is consistent with the currently known barrier results (i.e., the relativization barriers in [Imp11] and Section 9.1 and the barrier against nonadaptive black-box reductions [BT06b]) because our reduction from worst-case learning to average-case NP in Section 3.1 passes through a non-black-box technique [Hir18; Hir20b; HW20], and the proof of Theorem 2.6.5 is provably non-relativizing. Any progress on this approach (e.g., improving the approximation factor in the NP-hardness result and finding new barrier results) is interesting and important for excluding Heuristica from the perspective of learning.

Another important open problem is whether non-trivial distribution-free learning is possible in relativized Heuristica.

Open Question. Can we show that $\mathsf{P/poly}$ (or $\mathsf{SIZE}[n]$) is weakly PAC learnable in $2^n/n^{\omega(1)}$ time under $\mathsf{DistNP} \subseteq \mathsf{AvgP}$ by a relativizing proof? Or, is there any relativization barrier even for such non-trivial learning in errorless Heuristica?

It is also interesting to find some learning-theoretic characterization of the average-case errorless complexity of NP. Remember that, in Section 3.2, we present the characterization result in the error-prone case by introducing conditional extrapolation. We conjecture that an errorless analogue of the characterization result holds, which is naturally stated as follows:

Conjecture. *The following are equivalent.*

1. $\mathsf{DistNP} \subseteq \mathsf{AvgBPP}$.
2. **(Errorless Conditional Extrapolation)** *For every samplable distribution families* $\{\mathcal{C}_n\}_{n\in\mathbb{N}}$ *and* $\{\mathcal{D}_n\}_{n\in\mathbb{N}}$ *over binary strings, there exists a probabilistic polynomial-time algorithm* Ext *such that for all* $n, k, \epsilon^{-1}, \delta^{-1} \in \mathbb{N}$,

$$\Pr_{x\sim\mathcal{C}_n}\left[\Pr_{\mathsf{Ext}}[\mathsf{Ext}(x; 1^{\langle k,\epsilon^{-1},\delta^{-1}\rangle}) = \bot] > 1/4\right] \leq \delta.$$

and for every $x \in \mathrm{supp}(\mathcal{D}_n)$

$$\Pr_{\mathsf{Ext}}[\mathsf{Ext}(x; 1^{\langle k,\epsilon^{-1},\delta^{-1}\rangle}) = \bot] \leq 1/4 \Longrightarrow \mathsf{L}_1\left(\mathsf{Ext}(x; 1^{\langle k,\epsilon^{-1},\delta^{-1}\rangle}), \mathsf{Next}_k(x, \mathcal{D}_n)\right) \leq \epsilon.$$

The proof of Theorem 3.2.1 is needed to be modified to show the conjecture above because, in the proof of Theorem 3.2.1, we need to execute a heuristic scheme for the search version of the circuit SAT problem even on distributions on which the correctness of the heuristic scheme is not guaranteed. However, we believe that changing the distribution in the proof to one on which the heuristic scheme actually works is fine to show the conjecture above.

Quest for Error-Prone Heuristica

In Section 9.2, we present the relativization barrier against extending the currently known algorithmic implications from the average-case errorless easiness of NP to the case of the average-case error-prone easiness of NP. An important open question is to develop a non-relativized technique to overcome the barrier result.

We also remark that several important questions remain on relativizing techniques. The oracle separation in Section 9.2 heavily relies on the characteristics of DNFs (i.e., nondeterministic machines). Currently, it is unclear whether the proof proposed in Section 9.2 can be extended to a general case of constant-depth circuits, even for depth-3 $\wedge$-$\vee$-$\wedge$-circuits (which corresponds to Π^p_2). By contrast, the oracle separation between the worst-case hardness and the errorless average-case easiness for NP in [Imp11] is naturally extended for PH, as explicitly discussed in Section 9.1 by considering the switching lemma for constant-depth circuits. Therefore, we pose the following open question for the further research on the difference between the errorless and error-prone average-case complexity.

Open Question. Is there any oracle $\mathcal{O}$ relative to which $\mathsf{DistNP}^{\mathcal{O}} \not\subseteq \mathsf{AvgP}^{\mathcal{O}}/\mathsf{poly}$ and $\mathsf{DistPH}^{\mathcal{O}} \subseteq \mathsf{HeurP}^{\mathcal{O}}$? Or, is there a relativizing proof which shows that $\mathsf{DistPH} \subseteq \mathsf{HeurP} \Longrightarrow \mathsf{DistNP} \subseteq \mathsf{AvgP/poly}$?

The fact that we failed to extend our results to PH might suggest the feasibility of proving $\mathsf{DistPH} \subseteq \mathsf{HeurP} \implies \mathsf{DistNP} \subseteq \mathsf{AvgP/poly}$. Furthermore, we failed to improve our lower bound $2^{o(n/\log n)}$ on the time complexity of errorless average-case algorithms to $2^{o(n)}$. In this light, we conjecture that the worst-case-to-average-case connection of Hirahara [Hir21b], which shows that $\mathsf{DistNP} \subseteq \mathsf{AvgP} \implies \mathsf{UP} \subseteq \mathsf{DTIME}(2^{O(n/\log n)})$, can be extended to the error-prone average-case complexity by using a relativizing proof.

Conjecture. *For every oracle $\mathcal{O}$, if $\mathsf{DistNP}^{\mathcal{O}} \subseteq \mathsf{HeurP}^{\mathcal{O}}$, then $\mathsf{UP}^{\mathcal{O}} \subseteq \mathsf{BPTIME}^{\mathcal{O}}[2^{O(n/\log n)}]$.*

Another interesting open question is whether a unified and robust theory of learning can be established based on the conditional extrapolation (Theorem 3.2.1) similarly as the unified theory based on universal extrapolation in Chapter 4.

Quest for Pessiland

An important future direction is to investigate learning-theoretic consequences of the non-existence of OWF that are not directly derived from Theorem 4.3.3. Particularly, an important example is MINLT. By the NP-hardness result of MINLT (Theorem 2.6.5), proving the feasibility of MINLT in the average-case setting in Theorem 4.2.2 under the non-existence of OWF implies *excluding Pessiland*. This approach seems hopeful at present because of the non-relativizing proof in [Hir22a], and such a breakthrough does not contradict our current knowledge. Generally, the description length of hypotheses our learner produces heavily relies on the sample complexity. Therefore, as in Theorem 4.3.4, improving the sample complexity (depending on the minimum description length of the consistent hypothesis in this case) can be one approach for excluding Pessiland from learning theory. Another approach is to generalize the feasibility of MINLT in the BFKL model (Theorem 5.1.3) to more general average-case setting. It is also important to remove the additional derandomization assumption in Theorem 5.1.3.

Another important question is to show reductions similar to Corollary 4.2.3 in more restricted classes such as NC^1 and $\mathsf{AC}^0[p]$ instead of $\mathsf{P/poly}$. In computational learning theory, several novel learners have been developed for restrictive classes such as DNFs and $\mathsf{AC}^0[p]$, but almost all the learners require membership queries and work only under the uniform example distribution [LMN93; KM93; Jac97; CIKK16; CIKK17]. By showing a result similar to Corollary 4.2.3 for such restrictive classes, we can change the previous (somewhat theoretical) learners into more capable and practical learners.

Other important question is to establish the robust learning theory in the case in which samples are selected an unknown $\mathsf{P/poly}$-samplable distribution (i.e., worst-case with respect to $\mathsf{P/poly}$-samplable distributions) under the non-existence of *auxiliary-input* one-way functions.

Open Question. Can we show the equivalence of the following by a relativizing proof? Or, is there any oracle separation (i.e., the barrier against relativizing proofs)?

- The non-existence of auxiliary-input one-way function;
- Weak learning for $\mathsf{P/poly}$ with membership queries under all unknown $\mathsf{P/poly}$-samplable distributions over samples;
- Agnostic learning under all unknown $\mathsf{P/poly}$-samplable distributions over samples;
- Distributional learning for all unknown $\mathsf{P/poly}$-samplable distributions.

Note that either result, i.e., the equivalence or the oracle separation, will provide important knowledge. Particularly, the former yields new clear insights into the dichotomy between learning theory and cryptography along with this work, while the latter shows that the robustness of learning in Corollary 4.2.3 is indeed the unique property of average-case learning. We remark that Xiao [Xia10] presented a related oracle separation between weak learning and distributional learning, but they considered not efficiently samplable distributions (over examples) in weak learning.

Bibliography

[AB09] Sanjeev Arora and Boaz Barak. *Computational Complexity - A Modern Approach*. Cambridge University Press, 2009. ISBN: 978-0-521-42426-4.

[ABFKP08] Michael Alekhnovich, Mark Braverman, Vitaly Feldman, Adam R. Klivans, and Toniann Pitassi. "The complexity of properly learning simple concept classes". In: *J. Comput. Syst. Sci.* 74.1 (2008), pp. 16–34. DOI: 10.1016/j.jcss.2007.04.011.

[ABGKR14] Adi Akavia, Andrej Bogdanov, Siyao Guo, Akshay Kamath, and Alon Rosen. "Candidate weak pseudorandom functions in $\mathsf{AC}^0 \circ \mathsf{MOD}_2$". In: *Innovations in Theoretical Computer Science, ITCS'14, Princeton, NJ, USA, January 12-14, 2014*. 2014, pp. 251–260. DOI: 10.1145/2554797.2554821.

[ABR16] Benny Applebaum, Andrej Bogdanov, and Alon Rosen. "A Dichotomy for Local Small-Bias Generators". In: *J. Cryptol.* 29.3 (2016), pp. 577–596. DOI: 10.1007/s00145-015-9202-8. URL: https://doi.org/10.1007/s00145-015-9202-8.

[ABW10] Benny Applebaum, Boaz Barak, and Avi Wigderson. "Public-key cryptography from different assumptions". In: *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*. Ed. by Leonard J. Schulman. ACM, 2010, pp. 171–180. DOI: 10.1145/1806689.1806715. URL: https://doi.org/10.1145/1806689.1806715.

[ABX08] Benny Applebaum, Boaz Barak, and David Xiao. "On Basing Lower-Bounds for Learning on Worst-Case Assumptions". In: *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*. 2008, pp. 211–220. DOI: 10.1109/FOCS.2008.35.

[AC15] Dana Angluin and Dongqu Chen. "Learning a Random DFA from Uniform Strings and State Information". In: *Algorithmic Learning Theory - 26th International Conference, ALT 2015, Banff, AB, Canada, October 4-6, 2015, Proceedings*. Ed. by Kamalika Chaudhuri, Claudio Gentile, and Sandra Zilles. Vol. 9355. Lecture Notes in Computer Science. Springer, 2015, pp. 119–133. DOI: 10.1007/978-3-319-24486-0_8. URL: https://doi.org/10.1007/978-3-319-24486-0%5C_8.

[ACMTV21] Eric Allender, Mahdi Cheraghchi, Dimitrios Myrisiotis, Harsha Tirumala, and Ilya Volkovich. "One-Way Functions and a Conditional Variant of MKTP". In: *Proceedings of the Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*. 2021, 7:1–7:19. DOI: 10.4230/LIPIcs.FSTTCS.2021.7.

[ACR98] Alexander E. Andreev, Andrea E. F. Clementi, and José D. P. Rolim. "A New General Derandomization Method". In: *J. ACM* 45.1 (1998), pp. 179–213. DOI: 10.1145/273865.273933.

[Adl78] Leonard M. Adleman. "Two Theorems on Random Polynomial Time". In: *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*. 1978, pp. 75–83. DOI: 10.1109/SFCS.1978.37.

[AF09] Luis Filipe Coelho Antunes and Lance Fortnow. "Worst-Case Running Times for Average-Case Algorithms". In: *Proceedings of the Conference on Computational Complexity (CCC)*. 2009, pp. 298–303. DOI: 10.1109/CCC.2009.12.

[AFMV06] Luis Antunes, Lance Fortnow, Dieter van Melkebeek, and N. V. Vinodchandran. "Computational depth: Concept and applications". In: *Theor. Comput. Sci.* 354.3 (2006), pp. 391–404. DOI: 10.1016/j.tcs.2005.11.033.

[AGGM06] Adi Akavia, Oded Goldreich, Shafi Goldwasser, and Dana Moshkovitz. "On basing one-way functions on NP-hardness". In: *Proceedings of the Symposium on Theory of Computing (STOC)*. 2006, pp. 701–710. DOI: 10.1145/1132516.1132614.

[AGMMM18] Eric Allender, Joshua A. Grochow, Dieter van Melkebeek, Cristopher Moore, and Andrew Morgan. "Minimum Circuit Size, Graph Isomorphism, and Related Problems". In: *SIAM J. Comput.* 47.4 (2018), pp. 1339–1372. DOI: 10.1137/17M1157970.

[AH91] William Aiello and Johan Håstad. "Statistical Zero-Knowledge Languages can be Recognized in Two Rounds". In: *J. Comput. Syst. Sci.* 42.3 (1991), pp. 327–345. DOI: 10.1016/0022-0000(91)90006-Q.

[AIK06] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. "Cryptography in NC^0". In: *SIAM J. Comput.* 36.4 (2006), pp. 845–888. DOI: 10.1137/S0097539705446950. URL: https://doi.org/10.1137/S0097539705446950.

[AIK08] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. "On Pseudorandom Generators with Linear Stretch in NC^0". In: *Comput. Complex.* 17.1 (2008), pp. 38–69. DOI: 10.1007/s00037-007-0237-6.

[AK19] Benny Applebaum and Eliran Kachlon. "Sampling Graphs without Forbidden Subgraphs and Unbalanced Expanders with Negligible Error". In: *60th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2019, Baltimore, Maryland, USA, November 9-12, 2019*. Ed. by David Zuckerman. IEEE Computer Society, 2019, pp. 171–179. DOI: 10.1109/FOCS.2019.00020. URL: https://doi.org/10.1109/FOCS.2019.00020.

[AK95] Dana Angluin and Michael Kharitonov. "When Won't Membership Queries Help?" In: *J. Comput. Syst. Sci.* 50.2 (1995), pp. 336–355. DOI: 10.1006/jcss.1995.1026.

[AKL09] Vikraman Arvind, Johannes Köbler, and Wolfgang Lindner. "Parameterized learnability of juntas". In: *Theor. Comput. Sci.* 410.47-49 (2009), pp. 4928–4936. DOI: 10.1016/j.tcs.2009.07.003.

[AL18] Benny Applebaum and Shachar Lovett. "Algebraic Attacks against Random Local Functions and Their Countermeasures". In: *SIAM J. Comput.* 47.1 (2018), pp. 52–79. DOI: 10.1137/16M1085942.

[ALMSS98] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. "Proof Verification and the Hardness of Approximation Problems". In: *J. ACM* 45.3 (1998), pp. 501–555. DOI: 10.1145/278298.278306. URL: https://doi.org/10.1145/278298.278306.

[App13] Benny Applebaum. "Pseudorandom Generators with Long Stretch and Low Locality from Random Local One-Way Functions". In: *SIAM J. Comput.* 42.5 (2013), pp. 2008–2037. DOI: 10.1137/120884857. URL: https://doi.org/10.1137/120884857.

[AR16] Benny Applebaum and Pavel Raykov. "Fast Pseudorandom Functions Based on Expander Graphs". In: *Theory of Cryptography - 14th International Conference, TCC 2016-B, Beijing, China, October 31 - November 3, 2016, Proceedings, Part I*. Ed. by Martin Hirt and Adam D. Smith. Vol. 9985. Lecture Notes in Computer Science. 2016, pp. 27–56. DOI: 10.1007/978-3-662-53641-4_2. URL: https://doi.org/10.1007/978-3-662-53641-4%5C_2.

[AS98] Sanjeev Arora and Shmuel Safra. "Probabilistic Checking of Proofs: A New Characterization of NP". In: *J. ACM* 45.1 (1998), pp. 70–122. DOI: 10.1145/273865.273901. URL: https://doi.org/10.1145/273865.273901.

[AW09] Scott Aaronson and Avi Wigderson. "Algebrization: A New Barrier in Complexity Theory". In: *TOCT* 1.1 (2009), 2:1–2:54. DOI: 10.1145/1490270.1490272.

[BB15] Andrej Bogdanov and Christina Brzuska. "On Basing Size-Verifiable One-Way Functions on NP-Hardness". In: *Proceedings of the Theory of Cryptography Conference (TCC)*. 2015, pp. 1–6. DOI: 10.1007/978-3-662-46494-6_1.

[BCGIKS21] Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, and Peter Scholl. "Low-Complexity Weak Pseudorandom Functions in $\mathtt{AC}^0[\mathtt{MOD2}]$". In: *Proceedings of the International Cryptology Conference (CRYPTO)*. 2021, pp. 487–516. DOI: 10.1007/978-3-030-84259-8_17.

[BCGL92] Shai Ben-David, Benny Chor, Oded Goldreich, and Michael Luby. "On the Theory of Average Case Complexity". In: *J. Comput. Syst. Sci.* 44.2 (1992), pp. 193–219. DOI: 10.1016/0022-0000(92)90019-F.

[BCKRS22] Eric Binnendyk, Marco Carmosino, Antonina Kolokolova, Ramyaa Ramyaa, and Manuel Sabin. "Learning with distributional inverters". In: *The 33rd International Conference of Algorithmic Learning Theory (ALT2022)*. Proceedings of Machine Learning Research. PMLR, 2022.

[BEHW87] Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K. Warmuth. "Occam's Razor". In: *Inf. Process. Lett.* 24.6 (1987), pp. 377–380. DOI: 10.1016/0020-0190(87)90114-1.

[BFKL93] Avrim Blum, Merrick L. Furst, Michael J. Kearns, and Richard J. Lipton. "Cryptographic Primitives Based on Hard Learning Problems". In: *Proceedings of the International Cryptology Conference (CRYPTO)*. 1993, pp. 278–291. DOI: 10.1007/3-540-48329-2_24.

[BFP05] Harry Buhrman, Lance Fortnow, and Aduri Pavan. "Some Results on Derandomization". In: *Theory Comput. Syst.* 38.2 (2005), pp. 211–227. DOI: 10.1007/s00224-004-1194-y.

[BGS75] Theodore P. Baker, John Gill, and Robert Solovay. "Relativizations of the P =? NP Question". In: *SIAM J. Comput.* 4.4 (1975), pp. 431–442. DOI: 10.1137/0204037.

[BHL12] Ido Ben-Eliezer, Rani Hod, and Shachar Lovett. “Random low-degree polynomials are hard to approximate”. In: *Comput. Complex.* 21.1 (2012), pp. 63–81. DOI: 10.1007/s00037-011-0020-6.

[BK95] Manuel Blum and Sampath Kannan. “Designing Programs that Check Their Work”. In: *J. ACM* 42.1 (1995), pp. 269–291. DOI: 10.1145/200836.200880.

[BL13] Andrej Bogdanov and Chin Ho Lee. “Limits of Provable Security for Homomorphic Encryption”. In: *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I*. Ed. by Ran Canetti and Juan A. Garay. Vol. 8042. Lecture Notes in Computer Science. Springer, 2013, pp. 111–128. DOI: 10.1007/978-3-642-40041-4_7. URL: https://doi.org/10.1007/978-3-642-40041-4%5C_7.

[BM84] Manuel Blum and Silvio Micali. “How to Generate Cryptographically Strong Sequences of Pseudo-Random Bits”. In: *SIAM J. Comput.* 13.4 (1984), pp. 850–864. DOI: 10.1137/0213053.

[BQ12] Andrej Bogdanov and Youming Qiao. “On the security of Goldreich’s one-way function”. In: *Comput. Complex.* 21.1 (2012), pp. 83–127. DOI: 10.1007/s00037-011-0034-0.

[BR93] Avrim Blum and Ronald L. Rivest. “Training a 3-Node Neural Network is NP-Complete”. In: *Machine Learning: From Theory to Applications - Cooperative Research at Siemens and MIT*. Ed. by Stephen Jose Hanson, Werner Remmele, and Ronald L. Rivest. Vol. 661. Lecture Notes in Computer Science. Springer, 1993, pp. 9–28. DOI: 10.1007/3-540-56483-7_20. URL: https://doi.org/10.1007/3-540-56483-7%5C_20.

[Bra83] Gilles Brassard. “Relativized cryptography”. In: *IEEE Trans. Inf. Theory* 29.6 (1983), pp. 877–893. DOI: 10.1109/TIT.1983.1056754. URL: https://doi.org/10.1109/TIT.1983.1056754.

[BT06a] Andrej Bogdanov and Luca Trevisan. “Average-Case Complexity”. In: *Foundations and Trends in Theoretical Computer Science* 2.1 (2006). DOI: 10.1561/0400000004.

[BT06b] Andrej Bogdanov and Luca Trevisan. “On Worst-Case to Average-Case Reductions for NP Problems”. In: *SIAM J. Comput.* 36.4 (2006), pp. 1119–1159. DOI: 10.1137/S0097539705446974.

[CDMRR18] Geoffroy Couteau, Aurélien Dupin, Pierrick Méaux, Mélissa Rossi, and Yann Rotella. “On the Concrete Security of Goldreich’s Pseudorandom Generator”. In: *IACR Cryptol. ePrint Arch.* 2018 (2018), p. 1162.

[CEMT14] James Cook, Omid Etesami, Rachel Miller, and Luca Trevisan. “On the One-Way Function Candidate Proposed by Goldreich”. In: *ACM Trans. Comput. Theory* 6.3 (2014), 14:1–14:35. DOI: 10.1145/2633602.

[CIKK16] Marco L. Carmosino, Russell Impagliazzo, Valentine Kabanets, and Antonina Kolokolova. “Learning Algorithms from Natural Proofs”. In: *Proceedings of the Conference on Computational Complexity (CCC)*. 2016, 10:1–10:24. DOI: 10.4230/LIPIcs.CCC.2016.10.

[CIKK17] Marco L. Carmosino, Russell Impagliazzo, Valentine Kabanets, and Antonina Kolokolova. "Agnostic Learning from Tolerant Natural Proofs". In: *Proceedings of the Approximation, Randomization, and Combinatorial Optimization (APPROX/RANDOM)*. 2017, 35:1–35:19. DOI: 10.4230/LIPIcs.APPROX-RANDOM.2017.35.

[CLN14] Parinya Chalermsook, Bundit Laekhanukit, and Danupon Nanongkai. "Pre-reduction Graph Products: Hardnesses of Properly Learning DFAs and Approximating EDP on DAGs". In: *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*. 2014, pp. 444–453. DOI: 10.1109/FOCS.2014.54.

[CM01] Mary Cryan and Peter Bro Miltersen. "On Pseudorandom Generators in NC". In: *Mathematical Foundations of Computer Science 2001, 26th International Symposium, MFCS 2001 Marianske Lazne, Czech Republic, August 27-31, 2001, Proceedings*. Ed. by Jirí Sgall, Ales Pultr, and Petr Kolman. Vol. 2136. Lecture Notes in Computer Science. Springer, 2001, pp. 272–284. DOI: 10.1007/3-540-44683-4_24. URL: https://doi.org/10.1007/3-540-44683-4%5C_24.

[Dan16] Amit Daniely. "Complexity theoretic limitations on learning halfspaces". In: *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*. Ed. by Daniel Wichs and Yishay Mansour. ACM, 2016, pp. 105–117. DOI: 10.1145/2897518.2897520. URL: https://doi.org/10.1145/2897518.2897520.

[DP12] Ivan Damgård and Sunoo Park. "Is Public-Key Encryption Based on LPN Practical?" In: *IACR Cryptol. ePrint Arch.* (2012), p. 699. URL: http://eprint.iacr.org/2012/699.

[DS16] Amit Daniely and Shai Shalev-Shwartz. "Complexity Theoretic Limitations on Learning DNF's". In: *Proceedings of the 29th Conference on Learning Theory, COLT 2016, New York, USA, June 23-26, 2016*. 2016, pp. 815–830.

[DV21] Amit Daniely and Gal Vardi. "From Local Pseudorandom Generators to Hardness of Learning". In: *Conference on Learning Theory, COLT 2021, 15-19 August 2021, Boulder, Colorado, USA*. 2021, pp. 1358–1394.

[Fel09] Vitaly Feldman. "Hardness of approximate two-level logic minimization and PAC learning with membership queries". In: *J. Comput. Syst. Sci.* 75.1 (2009), pp. 13–26. DOI: 10.1016/j.jcss.2008.07.007.

[Fel10] Vitaly Feldman. "Distribution-Specific Agnostic Boosting". In: *Innovations in Computer Science - ICS 2010, Tsinghua University, Beijing, China, January 5-7, 2010. Proceedings*. 2010, pp. 241–250.

[FF93] Joan Feigenbaum and Lance Fortnow. "Random-Self-Reducibility of Complete Sets". In: *SIAM J. Comput.* 22.5 (1993), pp. 994–1005. DOI: 10.1137/0222061.

[FGKP09] Vitaly Feldman, Parikshit Gopalan, Subhash Khot, and Ashok Kumar Ponnuswami. "On Agnostic Learning of Parities, Monomials, and Halfspaces". In: *SIAM J. Comput.* 39.2 (2009), pp. 606–645. DOI: 10.1137/070684914.

[FI19] Yuval Filmus and Ferdinand Ihringer. "Boolean constant degree functions on the slice are juntas". In: *Discret. Math.* 342.12 (2019). DOI: 10.1016/j.disc.2019.111614. URL: https://doi.org/10.1016/j.disc.2019.111614.

[Fil22] Yuval Filmus. "Junta threshold for low degree Boolean functions on the slice". In: *CoRR* abs/2203.04760 (2022). DOI: 10.48550/arXiv.2203.04760. arXiv: 2203.04760. URL: https://doi.org/10.48550/arXiv.2203.04760.

[For13] Lance Fortnow. *The Golden Ticket - P, NP, and the Search for the Impossible*. Princeton University Press, 2013. ISBN: 978-0-691-15649-1. URL: http://press.princeton.edu/titles/9937.html.

[For94] Lance Fortnow. "The Role of Relativization in Complexity Theory". In: *Bull. EATCS* 52 (1994), pp. 229–243.

[FS96] Yoav Freund and Robert E. Schapire. "Experiments with a New Boosting Algorithm". In: *Machine Learning, Proceedings of the Thirteenth International Conference (ICML '96), Bari, Italy, July 3-6, 1996*. 1996, pp. 148–156.

[GGM84] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. "On the Cryptographic Applications of Random Functions". In: *Advances in Cryptology, Proceedings of CRYPTO '84, Santa Barbara, California, USA, August 19-22, 1984, Proceedings*. Ed. by G. R. Blakley and David Chaum. Vol. 196. Lecture Notes in Computer Science. Springer, 1984, pp. 276–288. DOI: 10.1007/3-540-39568-7_22. URL: https://doi.org/10.1007/3-540-39568-7%5C_22.

[GGM86] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. "How to construct random functions". In: *J. ACM* 33.4 (1986), pp. 792–807. DOI: 10.1145/6490.6503.

[GKK08] Parikshit Gopalan, Adam Tauman Kalai, and Adam R. Klivans. "Agnostically learning decision trees". In: *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*. Ed. by Cynthia Dwork. ACM, 2008, pp. 527–536. DOI: 10.1145/1374376.1374451. URL: https://doi.org/10.1145/1374376.1374451.

[GKLO22] Halley Goldberg, Valentine Kabanets, Zhenjian Lu, and Igor Carboni Oliveira. "Probabilistic Kolmogorov Complexity with Applications to Average-Case Complexity". In: *37th Computational Complexity Conference, CCC 2022, July 20-23, 2022, Philadelphia, PA, USA*. Ed. by Shachar Lovett. Vol. 234. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022, 16:1–16:60. DOI: 10.4230/LIPIcs.CCC.2022.16. URL: https://doi.org/10.4230/LIPIcs.CCC.2022.16.

[GKS10] Parikshit Gopalan, Subhash Khot, and Rishi Saket. "Hardness of Reconstructing Multivariate Polynomials over Finite Fields". In: *SIAM J. Comput.* 39.6 (2010), pp. 2598–2621. DOI: 10.1137/070705258.

[GL89] Oded Goldreich and Leonid A. Levin. "A Hard-Core Predicate for all One-Way Functions". In: *Proceedings of the Symposium on Theory of Computing (STOC)*. 1989, pp. 25–32. DOI: 10.1145/73007.73010.

[GMW91] Oded Goldreich, Silvio Micali, and Avi Wigderson. "Proofs that Yield Nothing But Their Validity for All Languages in NP Have Zero-Knowledge Proof Systems". In: *J. ACM* 38.3 (1991), pp. 691–729. DOI: 10.1145/116825.116852. URL: https://doi.org/10.1145/116825.116852.

[GNW11] Oded Goldreich, Noam Nisan, and Avi Wigderson. "On Yao's XOR-Lemma". In: *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation*. Springer, 2011, pp. 273–301. DOI: 10.1007/978-3-642-22670-0_23.

[Gol01] Oded Goldreich. *The Foundations of Cryptography - Volume 1: Basic Techniques*. Cambridge University Press, 2001. ISBN: 0-521-79172-3. DOI: 10.1017/CBO9780511546891.

[Gol04] Oded Goldreich. *The Foundations of Cryptography - Volume 2: Basic Applications*. Cambridge University Press, 2004. ISBN: 0-521-83084-2. DOI: 10.1017/CBO9780511721656.

[Gol11a] Oded Goldreich. "Candidate One-Way Functions Based on Expander Graphs". In: *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation - In Collaboration with Lidor Avigad, Mihir Bellare, Zvika Brakerski, Shafi Goldwasser, Shai Halevi, Tali Kaufman, Leonid Levin, Noam Nisan, Dana Ron, Madhu Sudan, Luca Trevisan, Salil Vadhan, Avi Wigderson, David Zuckerman*. 2011, pp. 76–87. DOI: 10.1007/978-3-642-22670-0_10.

[Gol11b] Oded Goldreich. "In a World of P=BPP". In: *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation - In Collaboration with Lidor Avigad, Mihir Bellare, Zvika Brakerski, Shafi Goldwasser, Shai Halevi, Tali Kaufman, Leonid Levin, Noam Nisan, Dana Ron, Madhu Sudan, Luca Trevisan, Salil Vadhan, Avi Wigderson, David Zuckerman*. Ed. by Oded Goldreich. Vol. 6650. Lecture Notes in Computer Science. Springer, 2011, pp. 191–232. DOI: 10.1007/978-3-642-22670-0_20. URL: https://doi.org/10.1007/978-3-642-22670-0%5C_20.

[GOSSW11] Parikshit Gopalan, Ryan O'Donnell, Rocco A. Servedio, Amir Shpilka, and Karl Wimmer. "Testing Fourier Dimensionality and Sparsity". In: *SIAM J. Comput.* 40.4 (2011), pp. 1075–1100. DOI: 10.1137/100785429. URL: https://doi.org/10.1137/100785429.

[GR09] Venkatesan Guruswami and Prasad Raghavendra. "Hardness of Learning Halfspaces with Noise". In: *SIAM J. Comput.* 39.2 (2009), pp. 742–765. DOI: 10.1137/070685798. URL: https://doi.org/10.1137/070685798.

[GT00] Rosario Gennaro and Luca Trevisan. "Lower Bounds on the Efficiency of Generic Cryptographic Constructions". In: *41st Annual Symposium on Foundations of Computer Science, FOCS 2000, 12-14 November 2000, Redondo Beach, California, USA*. IEEE Computer Society, 2000, pp. 305–313. DOI: 10.1109/SFCS.2000.892119. URL: https://doi.org/10.1109/SFCS.2000.892119.

[GV08] Dan Gutfreund and Salil P. Vadhan. "Limitations of Hardness vs. Randomness under Uniform Reductions". In: *Proceedings of the Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM)*. 2008, pp. 469–482. DOI: 10.1007/978-3-540-85363-3_37.

[GV99] Oded Goldreich and Salil P. Vadhan. "Comparing Entropies in Statistical Zero Knowledge with Applications to the Structure of SZK". In: *Proceedings of the Conference on Computational Complexity (CCC)*. 1999, p. 54. DOI: 10.1109/CCC.1999.766262.

[GVW11] Oded Goldreich, Salil P. Vadhan, and Avi Wigderson. "Simplified Derandomization of BPP Using a Hitting Set Generator". In: *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation - In Collaboration with Lidor Avigad, Mihir Bellare, Zvika Brakerski, Shafi Goldwasser, Shai Halevi, Tali Kaufman, Leonid Levin, Noam Nisan, Dana Ron, Madhu Sudan, Luca Trevisan, Salil Vadhan, Avi Wigderson, David Zuckerman*. Ed. by Oded Goldreich. Vol. 6650. Lecture Notes in Computer Science. Springer, 2011, pp. 59–67. DOI: 10.1007/978-3-642-22670-0_8. URL: https://doi.org/10.1007/978-3-642-22670-0%5C_8.

[Hås86] Johan Håstad. "Computational limitations of small-depth circuits". PhD thesis. MIT, 1986.

[HILL99] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. "A Pseudorandom Generator from any One-way Function". In: *SIAM J. Comput.* 28.4 (1999), pp. 1364–1396. DOI: 10.1137/S0097539793244708.

[HILNO23] Shuichi Hirahara, Rahul Ilango, Zhenjian Lu, Mikito Nanashima, and Igor Carboni Oliveira. "A Duality Between One-Way Functions and Average-Case Symmetry of Information". In: *The 55th ACM Symposium on Theory of Computing (STOC 2023)*. 2023.

[Hir18] Shuichi Hirahara. "Non-black-box Worst-case to Average-case Reductions within NP". In: *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*. 2018, pp. 247–258.

[Hir20a] Shuichi Hirahara. "Characterizing Average-Case Complexity of PH by Worst-Case Meta-Complexity". In: *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*. 2020, pp. 50–60.

[Hir20b] Shuichi Hirahara. "Non-Disjoint Promise Problems from Meta-Computational View of Pseudorandom Generator Constructions". In: *Proceedings of the Computational Complexity Conference (CCC)*. 2020, 20:1–20:47. DOI: 10.4230/LIPIcs.CCC.2020.20.

[Hir21a] Shuichi Hirahara. "Average-Case Hardness of NP from Exponential Worst-Case Hardness Assumptions". In: *Electron. Colloquium Comput. Complex.* 28 (2021). To appear in STOC 2021, p. 58.

[Hir21b] Shuichi Hirahara. "Average-case hardness of NP from exponential worst-case hardness assumptions". In: *Proceedings of the Symposium on Theory of Computing (STOC)*. 2021, pp. 292–302. DOI: 10.1145/3406325.3451065.

[Hir22a] Shuichi Hirahara. "NP-Hardness of Learning Programs and Partial MCSP". In: *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS*. 2022.

[Hir22b] Shuichi Hirahara. "Symmetry of Information from Meta-Complexity". In: *37th Computational Complexity Conference, CCC 2022, July 20-23, 2022, Philadelphia, PA, USA*. Ed. by Shachar Lovett. Vol. 234. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022, 26:1–26:41. DOI: 10.4230/LIPIcs.CCC.2022.26. URL: https://doi.org/10.4230/LIPIcs.CCC.2022.26.

[HJLT96] Thomas R. Hancock, Tao Jiang, Ming Li, and John Tromp. "Lower Bounds on Learning Decision Lists and Trees". In: *Inf. Comput.* 126.2 (1996), pp. 114–122. DOI: 10.1006/inco.1996.0040.

[HKLM22] Max Hopkins, Daniel M. Kane, Shachar Lovett, and Gaurav Mahajan. "Realizable Learning is All You Need". In: *Conference on Learning Theory, 2-5 July 2022, London, UK*. Ed. by Po-Ling Loh and Maxim Raginsky. Vol. 178. Proceedings of Machine Learning Research. PMLR, 2022, pp. 3015–3069. URL: https://proceedings.mlr.press/v178/hopkins22a.html.

[HKLW88] David Haussler, Michael J. Kearns, Nick Littlestone, and Manfred K. Warmuth. "Equivalence of Models for Polynomial Learnability". In: *Proceedings of the First Annual Workshop on Computational Learning Theory, COLT '88, Cambridge, MA, USA, August 3-5, 1988*. Ed. by David Haussler and Leonard Pitt. ACM/MIT, 1988, pp. 42–55. URL: http://dl.acm.org/citation.cfm?id=93040.

[HMX10] Iftach Haitner, Mohammad Mahmoody, and David Xiao. "A New Sampling Protocol and Applications to Basing Cryptographic Primitives on the Hardness of NP". In: *Proceedings of the Conference on Computational Complexity (CCC)*. 2010, pp. 76–87. DOI: 10.1109/CCC.2010.17.

[HN21] Shuichi Hirahara and Mikito Nanashima. "On Worst-Case Learning in Relativized Heuristica". In: *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*. 2021, pp. 751–758. DOI: 10.1109/FOCS52979.2021.00078.

[HN22] Shuichi Hirahara and Mikito Nanashima. "Finding Errorless Pessiland in Error-Prone Heuristica". In: *37th Computational Complexity Conference, CCC 2022, July 20-23, 2022, Philadelphia, PA, USA*. Ed. by Shachar Lovett. Vol. 234. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022, 25:1–25:28. DOI: 10.4230/LIPIcs.CCC.2022.25. URL: https://doi.org/10.4230/LIPIcs.CCC.2022.25.

[HN23a] Shuichi Hirahara and Mikito Nanashima. "A Unified Theory of Average-Case Learning". In: *manuscript*. 2023.

[HN23b] Shuichi Hirahara and Mikito Nanashima. "Learning versus Pseudorandom Generators in Constant Parallel Time". In: *14th Innovations in Theoretical Computer Science Conference, ITCS 2023, January 10-13, 2023, MIT in Cambridge, Massachusetts*. Ed. by Yael Tauman Kalai. Vol. 251. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023, 10:1–10:18.

[Hol05] Thomas Holenstein. "Key agreement from weak bit agreement". In: *Proceedings of the Symposium on Theory of Computing (STOC)*. 2005, pp. 664–673. DOI: 10.1145/1060590.1060689.

[Hol06] Thomas Holenstein. "Pseudorandom Generators from One-Way Functions: A Simple Construction for Any Hardness". In: *Proceedings of the Theory of Cryptography Conference (TCC)*. 2006, pp. 443–461. DOI: 10.1007/11681878_23.

[HRV13] Iftach Haitner, Omer Reingold, and Salil P. Vadhan. "Efficiency Improvements in Constructing Pseudorandom Generators from One-Way Functions". In: *SIAM J. Comput.* 42.3 (2013), pp. 1405–1430. DOI: 10.1137/100814421.

[HS17] Shuichi Hirahara and Rahul Santhanam. “On the Average-Case Complexity of MCSP and Its Variants”. In: *Proceedings of the Computational Complexity Conference (CCC)*. 2017, 7:1–7:20. DOI: 10.4230/LIPIcs.CCC.2017.7.

[HS22] Shuichi Hirahara and Rahul Santhanam. “Errorless Versus Error-Prone Average-Case Complexity”. In: *13th Innovations in Theoretical Computer Science Conference, ITCS 2022, January 31 - February 3, 2022, Berkeley, CA, USA*. Ed. by Mark Braverman. Vol. 215. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022, 84:1–84:23. DOI: 10.4230/LIPIcs.ITCS.2022.84. URL: https://doi.org/10.4230/LIPIcs.ITCS.2022.84.

[Hut05] Marcus Hutter. *Universal Artificial Intelligence: Sequential Decisions based on Algorithmic Probability*. Berlin: Springer, 2005, 300 pages. ISBN: 3-540-22139-5. DOI: 10.1007/b138233. URL: http://www.hutter1.net/ai/uaibook.htm.

[HW20] Shuichi Hirahara and Osamu Watanabe. “On Nonadaptive Security Reductions of Hitting Set Generators”. In: *Proceedings of the Approximation, Randomization, and Combinatorial Optimization (APPROX/RANDOM)*. 2020, 15:1–15:14. DOI: 10.4230/LIPIcs.APPROX/RANDOM.2020.15.

[IKOS08] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. “Cryptography with constant computational overhead”. In: *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*. Ed. by Cynthia Dwork. ACM, 2008, pp. 433–442. DOI: 10.1145/1374376.1374438. URL: https://doi.org/10.1145/1374376.1374438.

[IL89] Russell Impagliazzo and Michael Luby. “One-way Functions are Essential for Complexity Based Cryptography (Extended Abstract)”. In: *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*. 1989, pp. 230–235. DOI: 10.1109/SFCS.1989.63483.

[IL90] Russell Impagliazzo and Leonid A. Levin. “No Better Ways to Generate Hard NP Instances than Picking Uniformly at Random”. In: *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*. 1990, pp. 812–821. DOI: 10.1109/FSCS.1990.89604.

[ILO20] Rahul Ilango, Bruno Loff, and Igor Carboni Oliveira. “NP-Hardness of Circuit Minimization for Multi-Output Functions”. In: *Proceedings of the Computational Complexity Conference (CCC)*. 2020, 22:1–22:36. DOI: 10.4230/LIPIcs.CCC.2020.22.

[Imp] Russell Impagliazzo. “Impagliazzo PhD Thesis”. PhD thesis.

[Imp11] Russell Impagliazzo. “Relativized Separations of Worst-Case and Average-Case Complexities for NP”. In: *Proceedings of the Conference on Computational Complexity (CCC)*. 2011, pp. 104–114. DOI: 10.1109/CCC.2011.34.

[Imp95] Russell Impagliazzo. “A Personal View of Average-Case Complexity”. In: *Proceedings of the Structure in Complexity Theory Conference*. 1995, pp. 134–147. DOI: 10.1109/SCT.1995.514853.

[IR89] Russell Impagliazzo and Steven Rudich. "Limits on the Provable Consequences of One-Way Permutations". In: *Proceedings of the 21st Annual ACM Symposium on Theory of Computing, May 14-17, 1989, Seattle, Washington, USA*. Ed. by David S. Johnson. ACM, 1989, pp. 44–61. DOI: 10.1145/73007.73012. URL: https://doi.org/10.1145/73007.73012.

[IRS21] Rahul Ilango, Hanlin Ren, and Rahul Santhanam. "Hardness on any Samplable Distribution Suffices: New Characterizations of One-Way Functions by Meta-Complexity". In: *Electron. Colloquium Comput. Complex.* 28 (2021), p. 82.

[IRS22] Rahul Ilango, Hanlin Ren, and Rahul Santhanam. "Robustness of average-case meta-complexity via pseudorandomness". In: *STOC '22: 54th Annual ACM SIGACT Symposium on Theory of Computing, Rome, Italy, June 20 - 24, 2022*. Ed. by Stefano Leonardi and Anupam Gupta. ACM, 2022, pp. 1575–1583. DOI: 10.1145/3519935.3520051. URL: https://doi.org/10.1145/3519935.3520051.

[IW01] Russell Impagliazzo and Avi Wigderson. "Randomness vs Time: Derandomization under a Uniform Assumption". In: *J. Comput. Syst. Sci.* 63.4 (2001), pp. 672–688. DOI: 10.1006/jcss.2001.1780.

[Jac97] Jeffrey C. Jackson. "An Efficient Membership-Query Algorithm for Learning DNF with Respect to the Uniform Distribution". In: *J. Comput. Syst. Sci.* 55.3 (1997), pp. 414–440. DOI: 10.1006/jcss.1997.1533.

[JLS21] Aayush Jain, Huijia Lin, and Amit Sahai. "Indistinguishability obfuscation from well-founded assumptions". In: *STOC'21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*. Ed. by Samir Khuller and Virginia Vassilevska Williams. ACM, 2021, pp. 60–73. DOI: 10.1145/3406325.3451093. URL: https://doi.org/10.1145/3406325.3451093.

[JLS22] Aayush Jain, Huijia Lin, and Amit Sahai. "Indistinguishability Obfuscation from LPN over $\mathbb{F}_p$, DLIN, and PRGs in NC^0". In: *Advances in Cryptology - EUROCRYPT 2022 - 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30 - June 3, 2022, Proceedings, Part I*. Ed. by Orr Dunkelman and Stefan Dziembowski. Vol. 13275. Lecture Notes in Computer Science. Springer, 2022, pp. 670–699. DOI: 10.1007/978-3-031-06944-4_23. URL: https://doi.org/10.1007/978-3-031-06944-4%5C_23.

[JLSW11] Jeffrey C. Jackson, Homin K. Lee, Rocco A. Servedio, and Andrew Wan. "Learning random monotone DNF". In: *Discret. Appl. Math.* 159.5 (2011), pp. 259–271. DOI: 10.1016/j.dam.2010.08.022. URL: https://doi.org/10.1016/j.dam.2010.08.022.

[JS05] Jeffrey C. Jackson and Rocco A. Servedio. "Learning Random Log-Depth Decision Trees under Uniform Distribution". In: *SIAM J. Comput.* 34.5 (2005), pp. 1107–1128. DOI: 10.1137/S0097539704444555. URL: https://doi.org/10.1137/S0097539704444555.

[JVV86] Mark Jerrum, Leslie G. Valiant, and Vijay V. Vazirani. "Random Generation of Combinatorial Structures from a Uniform Distribution". In: *Theor. Comput. Sci.* 43 (1986), pp. 169–188. DOI: 10.1016/0304-3975(86)90174-X.

[KC00] Valentine Kabanets and Jin-yi Cai. "Circuit minimization problem". In: *Proceedings of the Symposium on Theory of Computing (STOC)*. 2000, pp. 73–79. DOI: 10.1145/335305.335314.

[Kha93] Michael Kharitonov. "Cryptographic hardness of distribution-specific learning". In: *Proceedings of the Symposium on Theory of Computing (STOC)*. 1993, pp. 372–381. DOI: 10.1145/167088.167197.

[KK09] Adam Kalai and Varun Kanade. "Potential-Based Agnostic Boosting". In: *Advances in Neural Information Processing Systems 22: 23rd Annual Conference on Neural Information Processing Systems 2009. Proceedings of a meeting held 7-10 December 2009, Vancouver, British Columbia, Canada*. Ed. by Yoshua Bengio, Dale Schuurmans, John D. Lafferty, Christopher K. I. Williams, and Aron Culotta. Curran Associates, Inc., 2009, pp. 880–888. URL: https://proceedings.neurips.cc/paper/2009/hash/13f9896df61279c928f19721878fac41-Abstract.html.

[KKMS08] Adam Tauman Kalai, Adam R. Klivans, Yishay Mansour, and Rocco A. Servedio. "Agnostically Learning Halfspaces". In: *SIAM J. Comput.* 37.6 (2008), pp. 1777–1805. DOI: 10.1137/060649057. URL: https://doi.org/10.1137/060649057.

[KL18] Pravesh K. Kothari and Roi Livni. "Improper Learning by Refuting". In: *Proceedings of the Innovations in Theoretical Computer Science Conference (ITCS)*. 2018, 55:1–55:10. DOI: 10.4230/LIPIcs.ITCS.2018.55.

[KLW10] Adam R. Klivans, Homin K. Lee, and Andrew Wan. "Mansour's Conjecture is True for Random DNF Formulas". In: *COLT 2010 - The 23rd Conference on Learning Theory, Haifa, Israel, June 27-29, 2010*. Ed. by Adam Tauman Kalai and Mehryar Mohri. Omnipress, 2010, pp. 368–380. URL: http://colt2010.haifa.il.ibm.com/papers/COLT2010proceedings.pdf%5C#page=376.

[KM93] Eyal Kushilevitz and Yishay Mansour. "Learning Decision Trees Using the Fourier Spectrum". In: *SIAM J. Comput.* 22.6 (1993), pp. 1331–1348. DOI: 10.1137/0222080. URL: https://doi.org/10.1137/0222080.

[KMRRSS94] Michael J. Kearns, Yishay Mansour, Dana Ron, Ronitt Rubinfeld, Robert E. Schapire, and Linda Sellie. "On the learnability of discrete distributions". In: *Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing, 23-25 May 1994, Montréal, Québec, Canada*. Ed. by Frank Thomson Leighton and Michael T. Goodrich. ACM, 1994, pp. 273–282. DOI: 10.1145/195058.195155. URL: https://doi.org/10.1145/195058.195155.

[Ko91] Ker-I Ko. "On the Complexity of Learning Minimum Time-Bounded Turing Machines". In: *SIAM J. Comput.* 20.5 (1991), pp. 962–986. DOI: 10.1137/0220059.

[KS09] Adam R. Klivans and Alexander A. Sherstov. "Cryptographic hardness for learning intersections of halfspaces". In: *J. Comput. Syst. Sci.* 75.1 (2009), pp. 2–12. DOI: 10.1016/j.jcss.2008.07.008. URL: https://doi.org/10.1016/j.jcss.2008.07.008.

[KS11] Subhash Khot and Rishi Saket. "On the hardness of learning intersections of two halfspaces". In: *J. Comput. Syst. Sci.* 77.1 (2011), pp. 129–141. DOI: 10.1016/j.jcss.2010.06.010. URL: https://doi.org/10.1016/j.jcss.2010.06.010.

[KSS94] Michael J. Kearns, Robert E. Schapire, and Linda Sellie. "Toward Efficient Agnostic Learning". In: *Mach. Learn.* 17.2-3 (1994), pp. 115–141. DOI: 10.1007/BF00993468. URL: https://doi.org/10.1007/BF00993468.

[KT91] Ker-I Ko and Wen-Guey Tzeng. "Three Σ_2^p-Complete Problems in Computational Learning Theory". In: *Comput. Complex.* 1 (1991), pp. 269–310. DOI: 10.1007/BF01200064. URL: https://doi.org/10.1007/BF01200064.

[KV94a] Michael J. Kearns and Leslie G. Valiant. "Cryptographic Limitations on Learning Boolean Formulae and Finite Automata". In: *J. ACM* 41.1 (1994), pp. 67–95. DOI: 10.1145/174644.174647.

[KV94b] Michael J. Kearns and Umesh V. Vazirani. *An Introduction to Computational Learning Theory*. MIT Press, 1994. ISBN: 978-0-262-11193-5. URL: https://mitpress.mit.edu/books/introduction-computational-learning-theory.

[Lau83] Clemens Lautemann. "BPP and the Polynomial Hierarchy". In: *Inf. Process. Lett.* 17.4 (1983), pp. 215–217. DOI: 10.1016/0020-0190(83)90044-3. URL: https://doi.org/10.1016/0020-0190(83)90044-3.

[Lev86] Leonid A. Levin. "Average Case Complete Problems". In: *SIAM J. Comput.* 15.1 (1986), pp. 285–286. DOI: 10.1137/0215020.

[LFKN92] Carsten Lund, Lance Fortnow, Howard J. Karloff, and Noam Nisan. "Algebraic Methods for Interactive Proof Systems". In: *J. ACM* 39.4 (1992), pp. 859–868. DOI: 10.1145/146585.146605.

[LMN93] Nathan Linial, Yishay Mansour, and Noam Nisan. "Constant Depth Circuits, Fourier Transform, and Learnability". In: *J. ACM* 40.3 (1993), pp. 607–620. DOI: 10.1145/174130.174138.

[LO22] Zhenjian Lu and Igor Carboni Oliveira. "Theory and Applications of Probabilistic Kolmogorov Complexity". In: *Bull. EATCS* 137 (2022). URL: http://bulletin.eatcs.org/index.php/beatcs/article/view/700.

[LOZ22] Zhenjian Lu, Igor Carboni Oliveira, and Marius Zimand. "Optimal Coding Theorems in Time-Bounded Kolmogorov Complexity". In: *49th International Colloquium on Automata, Languages, and Programming, ICALP 2022, July 4-8, 2022, Paris, France*. Ed. by Mikolaj Bojanczyk, Emanuela Merelli, and David P. Woodruff. Vol. 229. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022, 92:1–92:14. DOI: 10.4230/LIPIcs.ICALP.2022.92. URL: https://doi.org/10.4230/LIPIcs.ICALP.2022.92.

[LP20] Yanyi Liu and Rafael Pass. "On One-way Functions and Kolmogorov Complexity". In: *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*. 2020, pp. 1243–1254.

[LP21a] Yanyi Liu and Rafael Pass. "A Note on One-way Functions and Sparse Languages". In: *Electron. Colloquium Comput. Complex.* (2021), p. 92.

[LP21b] Yanyi Liu and Rafael Pass. "Cryptography from sublinear-time average-case hardness of time-bounded Kolmogorov complexity". In: *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*. Ed. by Samir Khuller and Virginia Vassilevska Williams. ACM, 2021, pp. 722–735. DOI: 10.1145/3406325.3451121. URL: https://doi.org/10.1145/3406325.3451121.

[LP21c] Yanyi Liu and Rafael Pass. "On the Possibility of Basing Cryptography on EXP≠ BPP". In: *Proceedings of the International Cryptology Conference (CRYPTO)*. 2021, pp. 11–40. DOI: 10.1007/978-3-030-84242-0_2.

[LP22] Yanyi Liu and Rafael Pass. "On One-Way Functions from NP-Complete Problems". In: *37th Computational Complexity Conference, CCC 2022, July 20-23, 2022, Philadelphia, PA, USA*. Ed. by Shachar Lovett. Vol. 234. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022, 36:1–36:24. DOI: 10.4230/LIPIcs.CCC.2022.36. URL: https://doi.org/10.4230/LIPIcs.CCC.2022.36.

[LV16] Tianren Liu and Vinod Vaikuntanathan. "On Basing Private Information Retrieval on NP-Hardness". In: *Theory of Cryptography - 13th International Conference, TCC 2016-A, Tel Aviv, Israel, January 10-13, 2016, Proceedings, Part I*. Ed. by Eyal Kushilevitz and Tal Malkin. Vol. 9562. Lecture Notes in Computer Science. Springer, 2016, pp. 372–386. DOI: 10.1007/978-3-662-49096-9_16. URL: https://doi.org/10.1007/978-3-662-49096-9%5C_16.

[LV19] Ming Li and Paul M. B. Vitányi. *An Introduction to Kolmogorov Complexity and Its Applications, 4th Edition*. Texts in Computer Science. Springer, 2019. ISBN: 978-3-030-11297-4. DOI: 10.1007/978-3-030-11298-1.

[LV89] Ming Li and Paul M. B. Vitányi. "A Theory of Learning Simple Concepts Under Simple Distributions and Average Case Complexity for the Universal Distribution (Extended Abstract)". In: *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*. 1989, pp. 34–39. DOI: 10.1109/SFCS.1989.63452.

[MF98] Neri Merhav and Meir Feder. "Universal Prediction". In: *IEEE Trans. Inf. Theory* 44.6 (1998), pp. 2124–2147. DOI: 10.1109/18.720534. URL: https://doi.org/10.1109/18.720534.

[MU05] Michael Mitzenmacher and Eli Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, 2005. ISBN: 978-0-521-83540-4. DOI: 10.1017/CBO9780511813603. URL: https://doi.org/10.1017/CBO9780511813603.

[Nan20] Mikito Nanashima. "Extending Learnability to Auxiliary-Input Cryptographic Primitives and Meta-PAC Learning". In: *Conference on Learning Theory, COLT 2020, 9-12 July 2020, Virtual Event [Graz, Austria]*. 2020, pp. 2998–3029.

[Nan21a] Mikito Nanashima. "A Theory of Heuristic Learnability". In: *Conference on Learning Theory, COLT 2021, 15-19 August 2021, Boulder, Colorado, USA*. Ed. by Mikhail Belkin and Samory Kpotufe. Vol. 134. Proceedings of Machine Learning Research. PMLR, 2021, pp. 3483–3525. URL: http://proceedings.mlr.press/v134/nanashima21a.html.

[Nan21b] Mikito Nanashima. "On Basing Auxiliary-Input Cryptography on NP-Hardness via Nonadaptive Black-Box Reductions". In: *Proceedings of the Innovations in Theoretical Computer Science Conference (ITCS)*. 2021, 29:1–29:15. DOI: 10.4230/LIPIcs.ITCS.2021.29.

[NOV06] Minh-Huyen Nguyen, Shien Jin Ong, and Salil P. Vadhan. "Statistical Zero-Knowledge Arguments for NP from Any One-Way Function". In: *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*. 2006, pp. 3–14. DOI: 10.1109/FOCS.2006.71.

[NR06] Moni Naor and Guy N. Rothblum. "Learning to impersonate". In: *Machine Learning, Proceedings of the Twenty-Third International Conference (ICML 2006), Pittsburgh, Pennsylvania, USA, June 25-29, 2006*. Ed. by William W. Cohen and Andrew W. Moore. Vol. 148. ACM International Conference Proceeding Series. ACM, 2006, pp. 649–656. DOI: 10.1145/1143844.1143926. URL: https://doi.org/10.1145/1143844.1143926.

[NR99] Moni Naor and Omer Reingold. "Synthesizers and Their Application to the Parallel Construction of Pseudo-Random Functions". In: *J. Comput. Syst. Sci.* 58.2 (1999), pp. 336–375. DOI: 10.1006/jcss.1998.1618.

[NY19] Moni Naor and Eylon Yogev. "Bloom Filters in Adversarial Environments". In: *ACM Trans. Algorithms* 15.3 (2019), 35:1–35:30. DOI: 10.1145/3306193. URL: https://doi.org/10.1145/3306193.

[ODo14] Ryan O'Donnell. *Analysis of Boolean Functions*. Cambridge University Press, 2014. ISBN: 978-1-10-703832-5.

[OS17] Igor Carboni Oliveira and Rahul Santhanam. "Conspiracies Between Learning Algorithms, Circuit Lower Bounds, and Pseudorandomness". In: *Proceedings of the Computational Complexity Conference (CCC)*. 2017, 18:1–18:49. DOI: 10.4230/LIPIcs.CCC.2017.18.

[OST22] Igor Carboni Oliveira, Rahul Santhanam, and Roei Tell. "Expander-Based Cryptography Meets Natural Proofs". In: *Comput. Complex.* 31.1 (2022), p. 4. DOI: 10.1007/s00037-022-00220-x. URL: https://doi.org/10.1007/s00037-022-00220-x.

[Ost91] Rafail Ostrovsky. "One-Way Functions, Hard on Average Problems, and Statistical Zero-Knowledge Proofs". In: *Proceedings of the Structure in Complexity Theory Conference*. 1991, pp. 133–138. DOI: 10.1109/SCT.1991.160253.

[OW14] Ryan O'Donnell and David Witmer. "Goldreich's PRG: Evidence for Near-Optimal Polynomial Stretch". In: *IEEE 29th Conference on Computational Complexity, CCC 2014, Vancouver, BC, Canada, June 11-13, 2014*. IEEE Computer Society, 2014, pp. 1–12. DOI: 10.1109/CCC.2014.9. URL: https://doi.org/10.1109/CCC.2014.9.

[OW93] Rafail Ostrovsky and Avi Wigderson. "One-Way Fuctions are Essential for Non-Trivial Zero-Knowledge". In: *Proceedings of the Symposium on Theory of Computing (STOC)*. 1993, pp. 3–17. DOI: 10.1109/ISTCS.1993.253489.

[PV88] Leonard Pitt and Leslie G. Valiant. "Computational limitations on learning from examples". In: *J. ACM* 35.4 (1988), pp. 965–984. DOI: 10.1145/48014.63140.

[PW90] Leonard Pitt and Manfred K. Warmuth. "Prediction-Preserving Reducibility". In: *J. Comput. Syst. Sci.* 41.3 (1990), pp. 430–467. DOI: 10.1016/0022-0000(90)90028-J.

[Raz93] Alexander A. Razborov. "An Equivalence between Second Order Bounded Domain Bounded Arithmetic and First Order Bounded Arithmetic". In: *Arithmetic, Proof Theory and Computational Complexity*. Ed. by Peter Clote and Jan Krajícek. Oxford University Press, 1993.

[Reg09] Oded Regev. "On lattices, learning with errors, random linear codes, and cryptography". In: *J. ACM* 56.6 (2009), 34:1–34:40. DOI: 10.1145/1568318.1568324.

[Rom90] John Rompel. "One-Way Functions are Necessary and Sufficient for Secure Signatures". In: *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing, May 13-17, 1990, Baltimore, Maryland, USA*. Ed. by Harriet Ortiz. ACM, 1990, pp. 387–394. DOI: 10.1145/100216.100269. URL: https://doi.org/10.1145/100216.100269.

[RR97] Alexander A. Razborov and Steven Rudich. "Natural Proofs". In: *J. Comput. Syst. Sci.* 55.1 (1997), pp. 24–35. DOI: 10.1006/jcss.1997.1494.

[RS21] Hanlin Ren and Rahul Santhanam. "Hardness of KT Characterizes Parallel Cryptography". In: *Proceedings of the Computational Complexity Conference (CCC)*. 2021, 35:1–35:58. DOI: 10.4230/LIPIcs.CCC.2021.35.

[RS22] Hanlin Ren and Rahul Santhanam. "A Relativization Perspective on Meta-Complexity". In: *39th International Symposium on Theoretical Aspects of Computer Science, STACS 2022, March 15-18, 2022, Marseille, France (Virtual Conference)*. Ed. by Petra Berenbrink and Benjamin Monmege. Vol. 219. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022, 54:1–54:13. DOI: 10.4230/LIPIcs.STACS.2022.54. URL: https://doi.org/10.4230/LIPIcs.STACS.2022.54.

[RTV04] Omer Reingold, Luca Trevisan, and Salil P. Vadhan. "Notions of Reducibility between Cryptographic Primitives". In: *Proceedings of the Theory of Cryptography Conference (TCC)*. 2004, pp. 1–20. DOI: 10.1007/978-3-540-24638-1_1.

[San20] Rahul Santhanam. "Pseudorandomness and the Minimum Circuit Size Problem". In: *Proceedings of the Innovations in Theoretical Computer Science Conference (ITCS)*. 2020, 68:1–68:26. DOI: 10.4230/LIPIcs.ITCS.2020.68.

[SB14] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning - From Theory to Algorithms*. Cambridge University Press, 2014. ISBN: 978-1-10-705713-5. URL: http://www.cambridge.org/de/academic/subjects/computer-science/pattern-recognition-and-machine-learning/understanding-machine-learning-theory-algorithms.

[Sch90] Robert E. Schapire. "The Strength of Weak Learnability". In: *Mach. Learn.* 5 (1990), pp. 197–227. DOI: 10.1007/BF00116037.

[Sch99] Marcus Schaefer. "Deciding the Vapnik-Červonenkis Dimension is Σ_3^p-Complete". In: *J. Comput. Syst. Sci.* 58.1 (1999), pp. 177–182. DOI: 10.1006/jcss.1998.1602. URL: https://doi.org/10.1006/jcss.1998.1602.

[Sel08] Linda Sellie. "Learning Random Monotone DNF Under the Uniform Distribution". In: *21st Annual Conference on Learning Theory - COLT 2008, Helsinki, Finland, July 9-12, 2008*. Ed. by Rocco A. Servedio and Tong Zhang. Omnipress, 2008, pp. 181–192. URL: http://colt2008.cs.helsinki.fi/papers/76-Sellie.pdf.

[Sel09] Linda Sellie. "Exact learning of random DNF over the uniform distribution". In: *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009*. Ed. by Michael Mitzenmacher. ACM, 2009, pp. 45–54. DOI: 10.1145/1536414.1536424. URL: https://doi.org/10.1145/1536414.1536424.

[Sha92] Adi Shamir. "IP = PSPACE". In: *J. ACM* 39.4 (1992), pp. 869–877. DOI: 10.1145/146585.146609. URL: https://doi.org/10.1145/146585.146609.

[Sol64a] Ray J. Solomonoff. "A Formal Theory of Inductive Inference. Part I". In: *Inf. Control.* 7.1 (1964), pp. 1–22. DOI: 10.1016/S0019-9958(64)90223-2.

[Sol64b] Ray J. Solomonoff. "A Formal Theory of Inductive Inference. Part II". In: *Inf. Control.* 7.2 (1964), pp. 224–254. DOI: 10.1016/S0019-9958(64)90131-7.

[SU05] Ronen Shaltiel and Christopher Umans. "Simple extractors for all min-entropies and a new pseudorandom generator". In: *J. ACM* 52.2 (2005), pp. 172–216. DOI: 10.1145/1059513.1059516.

[Tre10] Luca Trevisan. "The Program-Enumeration Bottleneck in Average-Case Complexity Theory". In: *Proceedings of the 25th Annual IEEE Conference on Computational Complexity, CCC 2010, Cambridge, Massachusetts, USA, June 9-12, 2010*. IEEE Computer Society, 2010, pp. 88–95. DOI: 10.1109/CCC.2010.18. URL: https://doi.org/10.1109/CCC.2010.18.

[Vad06] Salil P. Vadhan. "An Unconditional Study of Computational Zero Knowledge". In: *SIAM J. Comput.* 36.4 (2006), pp. 1160–1214. DOI: 10.1137/S0097539705447207.

[Vad17] Salil P. Vadhan. "On Learning vs. Refutation". In: *Proceedings of the 30th Conference on Learning Theory, COLT 2017, Amsterdam, The Netherlands, 7-10 July 2017*. 2017, pp. 1835–1848.

[Val84] Leslie G. Valiant. "A Theory of the Learnable". In: *Commun. ACM* 27.11 (1984), pp. 1134–1142. DOI: 10.1145/1968.1972.

[VZ12] Salil P. Vadhan and Colin Jia Zheng. "Characterizing pseudoentropy and simplifying pseudorandom generator constructions". In: *Proceedings of the Symposium on Theory of Computing (STOC)*. 2012, pp. 817–836. DOI: 10.1145/2213977.2214051.

[VZ13] Salil P. Vadhan and Colin Jia Zheng. "A Uniform Min-Max Theorem with Applications in Cryptography". In: *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I*. 2013, pp. 93–110. DOI: 10.1007/978-3-642-40041-4_6.

[Wat12] Thomas Watson. "Relativized Worlds without Worst-Case to Average-Case Reductions for NP". In: *TOCT* 4.3 (2012), 8:1–8:30. DOI: 10.1145/2355580.2355583.

[Wee06] Hoeteck Wee. "Finding Pessiland". In: *Proceedings of the Theory of Cryptography Conference (TCC)*. 2006, pp. 429–442. DOI: 10.1007/11681878_22.

[Wig19] Avi Wigderson. *Mathematics and Computation: A Theory Revolutionizing Technology and Science*. Princeton University Press, 2019. ISBN: 9780691192543.

[Xia09a] D. Xiao. "New Perspectives on the Complexity of Computational Learning, and Other Problems in Theoretical Computer Science". PhD thesis. Princeton University, 2009.

[Xia09b] David Xiao. "On Basing ZK $\not\subseteq$ BPP on the Hardness of PAC Learning". In: *Proceedings of the Conference on Computational Complexity (CCC)*. 2009, pp. 304–315. DOI: 10.1109/CCC.2009.11.

[Xia10] David Xiao. "Learning to Create is as Hard as Learning to Appreciate". In: *COLT 2010 - The 23rd Conference on Learning Theory, Haifa, Israel, June 27-29, 2010*. Ed. by Adam Tauman Kalai and Mehryar Mohri. Omnipress, 2010, pp. 516–528. URL: http://colt2010.haifa.il.ibm.com/papers/COLT2010proceedings.pdf%5C#page=524.

[Yao82] Andrew Chi-Chih Yao. "Theory and Applications of Trapdoor Functions (Extended Abstract)". In: *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*. 1982, pp. 80–91. DOI: 10.1109/SFCS.1982.45.

[Zim98] Marius Zimand. "Efficient privatization of random bits". In: *Proceedings of the International Symposium on Mathematical Foundations of Computer Science (MFCS)*. 1998.