ECCC

On the Power of Randomized Reductions and the Checkability of SAT

Mohammad Mahmoody[*] and David Xiao[†]

December 30, 2009

Abstract

The closure of complexity classes is a delicate question and the answer varies depending on the type of reduction considered. The closure of most classes under many-to-one (Karp) reductions is clear, but the question becomes complicated when oracle (Cook) reductions are allowed, and even more so when the oracle reductions are allowed to be randomized.

We first prove that $\mathbf{BPP^{SZK}} \subseteq \mathbf{AM} \cap \mathbf{coAM}$, strengthening the previously known facts that $\mathbf{SZK}$ is closed under $\mathbf{NC^1}$ truth-table reductions (Sahai and Vadhan, J. ACM '03) and that $\mathbf{P^{SZK}} \subseteq \mathbf{AM} \cap \mathbf{coAM}$ (Vadhan, as cited in Goldreich, ECCC '05). Our proof relies on showing that a certain class of real-valued functions that we call $\mathbb{R}\text{-}\mathbf{TFAM}$ can be computed using an $\mathbf{AM}$ protocol.

Then we investigate the power of randomized oracle reductions with relation to the notion of program checking (Blum and Kannan, J. ACM '95). We observe that a theorem of Beigel implies that if any problem in $\mathbf{TFNP}$ such as Nash equilibrium is $\mathbf{NP}$-hard under randomized oracle reductions, then SAT is checkable.

We also observe that Beigel's theorem can be extended to an average-case setting by relating checking to the notion of program *testing* (Blum et al., JCSS '93). From this, we derive that if one-way functions can be based on $\mathbf{NP}$-hardness via a randomized oracle reduction, then SAT is checkable. By showing that $\mathbf{NP}$ has a *non-uniform* tester, we also show that if SAT has a worst-case to average-case randomized oracle reduction, then SAT is checkable with a non-uniform instance checker. These results hold even for adaptive randomized oracle reductions.

1   Introduction

Studying the closure properties of complexity classes gives us insight into the power and behavior of those classes. For example, efficient classes such as $\mathbf{P}, \mathbf{BPP}$ are closed under efficient reductions to themselves: $\mathbf{P} = \mathbf{P}^{\mathbf{P}}$ and $\mathbf{BPP} = \mathbf{P}^{\mathbf{BPP}} = \mathbf{BPP}^{\mathbf{BPP}}$. More powerful classes however behave in more complicated ways. For example $\mathbf{NP}$ is not believed to be closed even under simple negation.

In this paper we study the class of randomized oracle (i.e., Cook) reductions: we say that a language L has a randomized oracle reduction to L', if there is an oracle algorithm A that, if given input x and oracle access to $\mathcal{O}$ that decides L' correctly on all inputs, then A correctly decides all inputs x with overwhelming probability over the random coins of the reduction. A can use the answers obtained from $\mathcal{O}$ in an arbitrary way (possibly even adaptively generating new queries to $\mathcal{O}$ using the answers it obtained from previous queries).

We will be interested in the behavior of promise problems under randomized oracle reductions. Promise problems (introduced by Even et al. [11], see also Goldreich [16]) are generalizations of languages: a promise problem Π consists of a set of YES instances Π_Y and a set of NO instances Π_N, where $\Pi_Y \cap \Pi_N = \varnothing$. Languages are just promise problems where $\Pi_N = \overline{\Pi}_Y$. For many basic complexity classes (notably $\mathbf{BPP}, \mathbf{AM}, \mathbf{SZK}$) it is actually more natural to consider them as classes of promise problems rather than classes of languages. This is because these classes are "semantic" and assume that their inputs satisfy certain properties: for example, we assume that a $\mathbf{BPP}$ algorithm accepts all inputs either with probability $\geq 2/3$ or $\leq 1/3$ (and never between $1/3$ and $2/3$), but we cannot check by looking at the algorithm only whether this promise holds for all the inputs or not.

We ask whether randomized oracle reductions using an oracle deciding a promise problem Π preserves certain properties of Π, such as complexity and checkability. Oracle reductions between promise problems can behave counter-intuitively: as pointed out by Even et al. [11], there is a (deterministic) oracle reduction A that can decide SAT using only an oracle which decides a promise problem in $\mathbf{NP} \cap \mathbf{co\text{-}NP}$. This is counter-intuitive, because if the oracle had decided a *language* in $\mathbf{NP} \cap \mathbf{co\text{-}NP}$ rather than a promise problem, then the existence of such a reduction A would have implied $\mathbf{NP} \subseteq \mathbf{co\text{-}NP}$, contradicting the widely held conjecture that $\mathbf{NP} \not\subseteq \mathbf{co\text{-}NP}$.

The reason that promise problems can behave in such strange ways under oracle reductions is because the oracle reductions can ask queries that fail the promise, and somehow gain additional power in doing so. Even et al. [11] show that if a reduction only asks the oracle on instances that satisfy the promise (they call such reductions "smart", which is arguably a misnomer since adding this restriction makes the reduction less powerful), then indeed any promise problem decidable using such a reduction with oracle access to a problem in $\mathbf{NP} \cap \mathbf{co\text{-}NP}$ must also be in $\mathbf{NP} \cap \mathbf{co\text{-}NP}$.

Goldreich [16] takes a different approach: they propose a restriction on *the structure of the problem* Π itself, and show that any language decidable using an arbitrary (though deterministic) oracle reduction with access to such a restricted problem must also lie in $\mathbf{NP} \cap \mathbf{co\text{-}NP}$.

We take this second approach in this paper, and show that if Π belongs to certain restricted classes of promise problems with "nice properties", then we are able to argue about either the complexity or the checkability of languages that reduce to Π under *randomized* oracle reductions.

1.1   Complexity of real-valued functions verifiable in AM

Our first main theorem studies the power of randomized oracle reductions that use oracle access to a class of real-valued function that we call $\mathbb{R}\text{-}\mathbf{TFAM}$. To understand $\mathbb{R}\text{-}\mathbf{TFAM}$, we begin by

discussing the well-known class **TFNP** [26]. The class **TFNP** is the following class of search problems: given input x, find y such that (x, y) satisfy a relation R, with the condition that R is efficiently decidable and R is *total*: for every x, there exists y such that $(x, y) \in R$.

TFAM is a natural relaxation of **TFNP** (see Definition 2.5). We still require that R is total, but now we allow R also to be just verifiable in **AM** (and not necessarily decidable in **P**). For functions where in addition $y \in \mathbb{R}$ and y is unique up to some small error, we say that the function is in $\mathbb{R}$-**TFAM** (see Definition 3.1).

Although to the best of our knowledge this class of functions is not well-studied, many natural problems can be decided given a $\mathbb{R}$-**TFAM** oracle. For example, consider the problem of Entropy Difference ED, which is complete for the class **SZK** [18]. An instance of this problem is a pair of circuits (X_1, X_2) that we think of them as samplers: the distribution sampled by the circuit $X : \{0, 1\}^m \mapsto \{0, 1\}^n$ is given by the output distribution $X_1(U_m)$ on uniform input bits. We write $H(X)$ to denote the Shannon entropy of the distribution sampled by X. A YES instance of ED satisfies $H(X_1) \geq H(X_2) + 1$, while a NO instance satisfies $H(X_2) \geq H(X_1) + 1$. It is clear that being able to evaluate the function $f(X_1, X_2) = H(X_1) - H(X_2)$ is sufficient to decide ED. It turns out that the entropy of distributions sampled by circuits can be verified using an **AM** protocol [14, 1] and therefore this function f is in the class $\mathbb{R}$-**TFAM**.

Our main theorem is about the limits of randomized oracle reductions with oracle access to languages in $\mathbb{R}$-**TFAM**.

Theorem 1.1. $\mathbf{BPP}^{\mathbb{R}\text{-}\mathbf{TFAM}} \subseteq \mathbf{AM} \cap \mathbf{coAM}$.

Application to SZK. It was previously known that **SZK** is closed under $\mathbf{NC^1}$ truth-table reductions [32], and also that $\mathbf{P^{SZK}} \subseteq \mathbf{AM} \cap \mathbf{coAM}$. The latter result was implicit in a result due to Vadhan, as cited in Theorem 5.4 of [16], and we sketch the proof in Section A. Theorem 1.1 and the above discussion showing that the **SZK**-complete problem ED can be decided using an $\mathbb{R}$-**TFAM** oracle says that one can strengthen Vadhan's result to the folloinwg.

Corollary 1.2. $\mathbf{BPP^{SZK}} \subseteq \mathbf{AM} \cap \mathbf{coAM}$.

Proof technique Theorem 1.1 is proven by constructing an **AM** protocol to "emulate" the behavior of a $\mathbb{R}$-**TFAM** oracle. The challenge in doing so is to prove the soundness against cheating provers. In particular, cheating provers can "adaptively" choose to answer queries in a way in order to trick the verifier. For example, a cheating prover might answer queries x_1, x_2 with y_1, y_2, but if the verifier asks the queries in reverse order x_2, x_1 then the prover might respond with y_2', y_1' where $y_2' \neq y_2$. Since a real $\mathbb{R}$-**TFAM** oracle would never behave in such a way, this kind of prover might cause the verifier to make mistakes. This is especially problematic for randomized oracle reductions because for each choice of the reduction's random coins, the prover may cheat in a different way. For the special case of $\mathbb{R}$-**TFAM** oracles we are able to add some noise to the prover's responses so that they are somewhat "independent" of the prover's choices, and this suffices to ensure that the verifier either catches the prover if it tries to cheat, or otherwise the verifier's emulation is correct.

1.2 Randomized oracle reductions, checkability, and testability

Although we are able to prove for the particular case of Theorem 1.1 that the verifier can catch the prover cheating, we do not know how to catch this kind of cheating in general. Therefore in the

following, we will *assume* that the cheating prover does not answer queries in such an adaptively malicious way. Namely, the prover's answers may be incorrect, but they are nevertheless a fixed function of the queries. The interesting point is that even achieving soundness only against such a restricted prover is tightly related to the notion of "checkability".

Checkability was defined by Blum and Kannan [6] and intuitively guarantees the following: C is a program checker for a problem Π if, when C is given an instance x and a program P, if P decides Π correctly (for all inputs, not just x), then $C(P, x)$ outputs the correct answer for x with high probability, while if P decides x incorrectly then with high probability $C(P, x)$ either outputs "error" or finds the correct answer despite the incorrectness of $P(x)$. Most checkers C require only oracle access to P and not the code of P.

It was already observed by Blum and Kannan [6] that L is checkable (with C using only oracle access to P) if and only if L and $\overline{L}$ have interactive proofs where the prover answers only queries of the form "$x \in L$?" and soundness is only required to hold against non-adaptively cheating provers of the kind discussed above.

In the following, we show that the **NP**-hardness of certain problems under randomized oracle reductions would imply that SAT is checkable. Whether or not SAT is checkable has been an open problem since checkability was defined [6]. Our results establish connections between the checkability of SAT and problems such as solving **TFNP** and inverting one-way functions, and give evidence that proving these latter problems to be **NP**-hard under randomized oracle reductions will be a challenging task as it would require resolving the open problem of whether SAT is checkable. We credit Holenstein [24] with the observation that connecting the hardness of various problems to the checkability of SAT is a meaningful statement.

1.2.1 An application of Beigel's theorem

Theorem 1.3 (Beigel, as cited in [6]). *Suppose Π is decidable by a randomized reduction A with oracle access to Π', and conversely Π' is decidable by a randomized reduction A' with oracle access to Π. Then Π is checkable if and only if Π' is checkable.*

By definition **TFNP** relations are checkable: given an oracle $\mathcal{O}$, check that $(x, \mathcal{O}(x)) \in R$ indeed holds. Because the relation is total, $\mathcal{O}(x)$ can never claim that no solution exists.

TFNP contains important problems such as Nash equilibrium [30]. Megiddo and Papadimitriou [26] observed that $\mathbf{P^{TFNP}} \subseteq \mathbf{NP} \cap \mathbf{co\text{-}NP}$ and therefore no **TFNP** oracle can be used to decide SAT under deterministic oracle reductions unless $\mathbf{NP} = \mathbf{co\text{-}NP}$. Consequently, it is unlikely that problems such as finding Nash equilibrium are **NP**-hard under deterministic oracle reductions. Since **TFNP** is trivially reducible to **NP**, as a corollary of Theorem 1.3 we observe the following about the hardness of **TFNP**:

Corollary 1.4. *If* $\mathrm{SAT} \in \mathbf{BPP^{TFNP}}$, *then* SAT *is checkable.*

As discussed above, one consequence is that if finding Nash equilibrium can be used to solve SAT via a *randomized* oracle reduction, then SAT is checkable. Corollary 1.4 can be interpreted as an incomparable version of the theorem of Megiddo and Papadimitriou [26], where our version handles randomized reductions but arrives at a different (weaker) conclusion.

1.2.2 Extending Beigel's theorem to testability

So far we have considered worst-case to worst-case reductions, but equally interesting are worst-case to average-case randomized oracle reductions. Such a reduction is an efficient oracle algorithm A that is guaranteed to decide Π correctly (on all inputs) given oracle access to $\mathcal{O}$ that decides Π' on average over an input distribution D, namely when $\Pr_{x \leftarrow D}[\mathcal{O}(x) \neq \Pi(x)] \leq 1/\operatorname{poly}(n)$.

A notion of checkability for average-case problems also exists, which following Blum et al. [8] we call testability: we say that C is a tester for Π' with respect to the input distribution D if the following holds. Whenever C is given a program P that correctly decides all x then C must also decide all x. If C is given access to P such that $\Pr_{x \leftarrow D}[P(x) \neq \Pi(x)] \geq \delta$ for some error parameter δ, then C outputs "error" with high probability. See Definition 2.7 for a formal definition. We remark that we allow C to run in time depending on the input length, while the related notion of property testing [8, 19] typically considers testing algorithms that run in time depending only on $1/\delta$. Also, typically C will only use oracle access to P.

Our first observation is that Beigel's theorem can be extended to encompass testability:

Theorem 1.5 (Extended Beigel's theorem, informal)**.** *Suppose Π is decidable by a worst-case to average-case randomized reduction A with oracle access to Π', and conversely Π' is decidable by a (worst-case to worst-case) randomized reduction A' with oracle access to Π. Then if Π' is testable, then both Π and Π' are checkable.*

Application to basing one-way functions on NP-hardness. One of the main goals of theoretical cryptography is to base the existence of cryptographic primitives on reasonable assumptions. $\mathbf{P} \neq \mathbf{NP}$ is a minimal assumption, and it would be ideal if one could construct one-way functions from this assumption. Such a result might be proved by giving an efficient reduction A that uses an oracle $\mathcal{O}$ that inverts a one-way function in order to decide SAT. However, it has been shown in a series of works [12, 9, 4, 31] that such a hope is most likely false if A is non-adaptive, since it would imply that $\mathbf{NP} \subseteq \mathbf{coAM}$.

We know less about whether or not one can base one-way functions on $\mathbf{NP}$-hardness via an *adaptive* reduction. There are certain cryptographic [22, 3, 27] and complexity-theoretic [13, 23, 5] settings where having the adaptivity in the reduction buys more power, and so it is important to understand whether adaptive reductions basing one-way functions on $\mathbf{NP}$-hardness are possible. Brassard [10] showed that one-way *permutations* cannot be based on $\mathbf{NP}$-hardness unless $\mathbf{NP} \subseteq \mathbf{coAM}$. Akavia et al. [4] generalized the result of Brassard [10] to one-way functions whose range is efficiently verifiable and whose preimage sizes are efficiently computable. Pass [31] showed that if (general) one-way functions can be based on $\mathbf{NP}$-hardness, then certain *witness-hiding* protocols do not exist. However Haitner et al. [21] showed that it is unlikely that known witness-hiding protocols are of the type studied by [31]. More recently, Haitner et al. [20] study the case of stronger primitives such as collision-resistant hash functions and constant-round statistically hiding commitments that can be broken by recursive collision finders.

Here we show that an adaptive reduction that uses a one-way function inverter to decide SAT would imply that SAT is checkable. The observation is that a one-way function is testable (for a suitable notion of testability for search problems), and therefore we can apply Theorem 1.5.

Corollary 1.6 (Joint with Holenstein)**.** *If there is a randomized oracle reduction that uses an oracle $\mathcal{O}$ which inverts a one-way function in order to decide SAT, then SAT is checkable.*

We note that the proof that was joint with Holenstein was direct and did not go through the formalism of testability that we develop here.

Application to worst-case to average-case reductions for SAT. Bogdanov and Trevisan [9] prove that non-adaptive worst-case to average-case randomized reductions for **NP** imply that **NP** $\subseteq$ **co-NP**$/$poly, which implies that **PH** $= \Sigma_3$ and is considered implausible. We will consider the same problem but allow adaptive reductions. Using a technique of [12] we observe that:

Theorem 1.7. *For any* $L \in$ **NP***, L is testable by a* non-uniform *tester.*

Theorem 4.3 states this formally. Combined with Theorem 1.5 this will imply the following.

Corollary 1.8. *Suppose there is a (possibly adaptive) worst-case to average-case reduction that uses an oracle $\mathcal{O}$ solving some* **NP** *language on average in order to decide* SAT *on all inputs. Then* SAT *is checkable with a non-uniform checker.*

1.3 Randomized vs. deterministic oracle reductions

As already noted throughout the introduction, if we restrict our attention to deterministic oracle reductions then all of the results we prove are already known (and indeed in most cases stronger conclusions hold). We remark that, although it is commonly conjectured that **P** $=$ **BPP** [35, 7, 28, 25], the techniques used to prove derandomization under commonly held hardness assumptions do not necessarily say that for some oracle $\mathcal{O}$, it holds that $\mathbf{P}^{\mathcal{O}} = \mathbf{BPP}^{\mathcal{O}}$. Indeed, there are examples of $\mathcal{O}$ where $\mathbf{P}^{\mathcal{O}} \neq \mathbf{BPP}^{\mathcal{O}}$. Therefore one cannot apply the general derandomization theorems above to the previously known results about, say, $\mathbf{P}^{\mathbf{SZK}}$ and $\mathbf{P}^{\mathbf{TFNP}}$ to derive our results. We therefore argue directly about $\mathbf{BPP}^{\mathbf{SZK}}$, $\mathbf{BPP}^{\mathbf{TFNP}}$, etc. without relying on derandomization assumptions.

2 Preliminaries

By $x \leftarrow X$ for random variable X we mean that x is sampled according to the distribution of X. For a set S, by U_S we mean the random variable with uniform distribution over S. By U_n we mean the random variable uniformly distributed over $\{0,1\}^n$.

As is standard convention, when we say an event occurs with negligible probability we mean it occurs with probability $n^{-\omega(1)} = \text{neg}(n)$ and if it occurs with overwhelming probability then it occurs with probability $1 - \text{neg}(n)$. Here, n is the input length. We call an algorithm *efficient* if it runs in time poly(n).

We assume that the reader is familiar with the idea of promise problems [11] as well as the standard complexity classes **BPP**, **NP**, **PH**, **SZK**, **AM**, etc., which we take to be classes of promise problems (and not just languages).

For every promise problem $\Pi = (\Pi_Y, \Pi_N)$ we write $\Pi(x) = 1$ if $x \in \Pi_Y$ and $\Pi(x) = 0$ if $x \in \Pi_N$. A language L is simply a promise problem $(L, \overline{L})$, where $\overline{L}$ is the complement of L.

To extend promise decision problems to search problems, we work with *promise relations*.

Definition 2.1 (Promise relations)**.** *Let $R = (R_Y, R_N)$ where $R_Y \subseteq \{0,1\}^* \times \{0,1\}^*$ and $R_N \subseteq \{0,1\}^* \times \{0,1\}^*$ such that $R_Y \cap R_N = \varnothing$. We call R a* promise relation *and sometimes write for shorthand $(x,y) \in R$ to mean $(x,y) \in R_Y$. We say R is a* standard *relation if $R_Y = \overline{R_N}$.*

For any promise relation $R = (R_Y, R_N)$, define $R(x) = \{y \mid (x, y) \in R_Y\}$. A relation R is *total* if for every x, $R(x) \neq \varnothing$. Note that the decision problem for total relations is trivial, but the search problem may still be hard.

In the following, we will consider $\varnothing$ to also denote a special symbol signifying the empty set, so that search algorithms are allowed to output $\varnothing$ to mean that the algorithm cannot find a solution. We will also abuse notation slightly and allow $A(x) \in R(x)$ to be a true statement if $R(x)$ is empty and $A(x)$ outputs the special symbol "$\varnothing$".

We say an algorithm A solves the language L (resp. the relation R) if for all x, it holds that $A(x) = L(x)$ (resp. $A(x) \in R(x)$). If the algorithm A is randomized, the latter should hold with overwhelming probability (for all x's).

Relations in NP and AM

Definition 2.2 (**NP** Relation)**.** *A standard relation $R \subseteq \{0,1\}^* \times \{0,1\}^*$ is an* **NP** *relation if the set R is accepted by an efficient algorithm, and for all $(x, y) \in R$, $|y| \leq \text{poly}(|x|)$.*

Definition 2.3 (**TFNP**)**.** *A standard relation $R \subseteq \{0,1\}^* \times \{0,1\}^*$ is in the class* **TFNP** *if R is an* **NP** *relation and R is total.*

Definition 2.4 (**AM** relations)**.** *For a promise relation $R = (R_Y, R_N)$, define the promise problem $M^R = \{M_Y^R, M_N^R\}$ where $M_Y^R = \{(x, y) \mid (x, y) \in R_Y\}$ and $M_N^R = \{(x, y) \mid (x, y) \in R_N\}$. R is an* **AM** *relation if $M^R \in$ **AM**, and also $\forall (x, y) \in R_Y$ it holds that $|y| \leq \text{poly}(|x|)$.*

Definition 2.5 (**TFAM**)**.** *A promise relation $R = (R_Y, R_N)$ is in* **TFAM** *if R is a total relation and also an* **AM** *relation.*

Since our definition is for promise problems, instances $(x, y) \notin R_Y \cup R_N$ can behave arbitrarily. It follows immediately from the definitions that **TFNP** $\subseteq$ **TFAM**.

When are AM relations interesting? In contrast to **NP**, not every **AM** decision problem has an interesting search version. This is because an **AM** decision problem might not have well-defined witnesses: for example, consider the standard Graph Non-Isomorphism (GNI) protocol, where given (G_1, G_2) the verifier picks random bit b and sends a random permutation of G_b to the prover, who must then respond with b. There is no fixed "witness" of the non-isomorphism, so the natural **AM** relation of Graph Non-Isomorphism is trivial, $((G_1, G_2), y) \in R_{\text{GNI}}$ iff $(G_1, G_2) \in$ GNI, and the value y is ignored. On the other hand, some interesting problems not known to be in **NP** (such as Entropy Difference ED) do have interesting witnesses that are only known to be verifiable using randomized protocols and not deterministically (for example, the entropy difference of a pair of circuits, see Section 3).

Checkability

Definition 2.6 (Checkability)**.** *A relation R is* checkable *if there exists an efficient randomized oracle algorithm A such that for all oracles $\mathcal{O}$, the following holds.*

- *Completeness: suppose for all x it holds that $\mathcal{O}(x) \in R(x)$. Then for all x, $A^{\mathcal{O}}(x) \in R(x)$ with overwhelming probability.*

- *Soundness: For any x such that $\mathcal{O}(x) \notin R(x)$, $A^{\mathcal{O}}(x)$ outputs with overwhelming probability either some $y \in R(x)$ or a special error symbol $\bot$.*

This definition coincides with the definition of [6] for checkability of promise problems Π if one considers the relation $(x, y) \in R$ iff $\Pi(x) = y$, where $y \in \{0, 1\}$.

It is known [6, 15] that for any $k \geq 2$ and any language L, L is checkable if and only if both L and $\overline{L}$ have an interactive proof system with k provers where the provers are asked only L queries.

An average case notion of checkability was defined by Blum et al. [8], which they called "program testing". (In the following we keep the convention that D_n is a distribution over $\{0, 1\}^n$.)

Definition 2.7 (Testability). *A relation R is δ-testable over the ensemble of distributions $D = \{D_n\}$ if there exists an efficient randomized oracle algorithm A such that for all oracles $\mathcal{O}$ the following holds.*

- *Completeness: If $\mathcal{O}(x) \in R(x)$ for all x, then $A^{\mathcal{O}}(1^n) = 1$ with overwhelming probability.*

- *Soundness: If $\Pr_{x \leftarrow D_n}[\mathcal{O}(x) \in R(x)] < \delta$, then $A^{\mathcal{O}}(1^n)$ outputs a special error symbol $\bot$ with overwhelming probability.*

As for checkability, the definition of program testing for decision problems Π follows immediately by considering the relation $(x, y) \in R$ iff $\Pi(x) = y$.

Worst-case to average-case reductions

For let D_n be a distribution over $\{0, 1\}^n$. We say that the ensemble of distributions $D = \{D_n\}$ is *samplable* if there is an efficient randomized algorithm S which the output of $S(1^n)$ is distributed according to D_n. For $\rho = \rho(n)$, we say that an oracle $\mathcal{O}$ ρ-solves the relation R over D if for every n it holds that $\Pr_{x \leftarrow D_n}[\mathcal{O}(x) \in R(x)] \geq \rho$.

Definition 2.8 (Worst-case to average-case reductions.). *Let $\rho = \rho(n)$, and let D be an ensemble of distributions. We say that the relation R reduces to ρ-solving the relation R' over D if there is an efficient randomized oracle algorithm A such that $\Pr[A^{\mathcal{O}}(x) \in R(x)] = 1 - \mathrm{neg}(n)$ for every $x \in \{0, 1\}^n$ whenever $\mathcal{O}$ ρ-solves the relation R' over D.*

Definition 2.9 (Worst-case to average-inverting reductions.). *Let $f = \{f_n \colon \{0, 1\}^n \to \{0, 1\}^n\}$ be a family of functions. We say that the relation R reduces to ρ-inverting f if R reduces to ρ-solving the relation R^f over the ensemble of distributions D^f where $R^f = \{(y, x) \colon f(x) = y\}$ and $D_n^f = f(U_n)$.*

Note that in both Definitions 2.8 and 2.9, the reduction is allowed to ask oracle queries y of larger length $|y| = \mathrm{poly}(|x|)$ where x is the input to the reduction.

3 Real-valued Total Functions

3.1 Definitions and Preliminaries

We begin by defining the class of relations $\mathbb{R}$-**TFAM**, which intuitively captures functions $f \colon \{0, 1\}^* \to \mathbb{R}$ such that given (x, y), it is possible to verify using an **AM** protocol that $|y - f(x)|$ is small.

Definition 3.1 (ℝ-**TFAM**). *A function $f : \{0,1\}^* \to \mathbb{R}$ is in ℝ-**TFAM** if for every $\varepsilon \geq 1/\operatorname{poly}(n)$, the following relation $R = (R_Y, R_N)$ is in* **AM**:

  1. $(x, f(x)) \in R_Y$.

  2. $(x, y) \in R_N$ *for all y such that $|y - f(x)| > \varepsilon$.*

We let $\mathbf{BPP}^{\mathbb{R}\text{-}\mathbf{TFAM}}$ denote the class of promise problems that are decidable by a randomized oracle algorithm given oracle access to a real-valued function whose output is verifiable in **AM**, formalized as follows.

Definition 3.2 ($\mathbf{BPP}^{\mathbb{R}\text{-}\mathbf{TFAM}}$). *$\Pi \in \mathbf{BPP}^{\mathbb{R}\text{-}\mathbf{TFAM}}$ if there exists an $f \in$ ℝ-**TFAM**, an oracle algorithm A, and an $\varepsilon = 1/\operatorname{poly}(n)$ such that for all oracles $\mathcal{O}$ satisfying $\forall x$, $|f(x) - \mathcal{O}(x)| \leq \varepsilon$, it holds for all $z \in \Pi_Y \cup \Pi_N$ that $A^{\mathcal{O}}(z) = \Pi(z)$ with overwhelming probability.*

Our main interest in studying ℝ-**TFAM** is its relationship to **SZK**. We will characterize **SZK** by its complete problem: Entropy Difference ED. See e.g., [34] for an introduction and other definitions of **SZK**.

Definition 3.3 (**SZK** and ED, [18]). *A promise problem $\Pi \in$ **SZK** if and only if it is Karp-reducible to the following promise problem $\mathsf{ED} = (\mathsf{ED}_Y, \mathsf{ED}_N)$. Instances of ED are pairs of circuits (X_1, X_2) where each circuit $X : \{0,1\}^m \mapsto \{0,1\}^n$ is identified by its output distribution $X(U_m)$. Let $H(X) = H(X(U_m))$ denote the Shannon entropy of $X(U_m)$. Then:*

  1. $(X_1, X_2) \in \mathsf{ED}_Y$ *iff $H(X_1) \geq H(X_2) + 1$.*

  2. $(X_1, X_2) \in \mathsf{ED}_N$ *iff $H(X_2) \geq H(X_1) + 1$.*

We first recall that the entropy can be estimated using an **AM** protocol, whose proof we give in the appendix.

Lemma 3.4 ([14, 2, 17]). *For every $\varepsilon > 1/\operatorname{poly}(n)$, there is an **AM** protocol that on input (X, y) accepts if $H(X) = y$ and rejects if $|H(X) - y| > \varepsilon$.*

Proposition 3.5. $\mathbf{BPP}^{\mathbf{SZK}} \subseteq \mathbf{BPP}^{\mathbb{R}\text{-}\mathbf{TFAM}}$

Proof. We show how to implement an ED oracle using a ℝ-**TFAM** oracle for any $\varepsilon \leq 1/2$. Consider the function $f(X_1, X_2) = H(X_1) - H(X_2)$. This function is in ℝ-**TFAM** because by Lemma 3.4 the entropy of a circuit can be approximated using an **AM** protocol.

We claim that given any oracle $\mathcal{O}$ that solves (the search problem of) the relation R with up to ε error, one can decide ED: given (X_1, X_2), query (X_1, X_2) from $\mathcal{O}$ to get $y = \mathcal{O}(X_1, X_2)$ and accept iff $y > 0$. Since we are guaranteed that $|(H(X_1) - H(X_2)) - y| \leq \varepsilon$, it therefore holds that if $H(X_1) > H(X_2) + 1$ then $y > 1 - \varepsilon$ and if $H(X_1) < H(X_2) - 1$ then $y < -1 + \varepsilon$. $\square$

3.2 Power of ℝ-TFAM

We now prove Theorem 1.1, namely $\mathbf{BPP}^{\mathbb{R}\text{-}\mathbf{TFAM}} \subseteq \mathbf{AM} \cap \mathbf{coAM}$.

Proof of Theorem 1.1. Fix any $\Pi \in \mathbf{BPP}^{\mathbb{R}\text{-}\mathbf{TFAM}}$, then by definition there exists $f \in \mathbb{R}\text{-}\mathbf{TFAM}$, an efficient oracle algorithm A, and a parameter $\varepsilon \geq 1/\operatorname{poly}(n)$ satisfying Definition 3.2. Define the (randomized) oracle $\mathcal{O}_\varepsilon$ as follows. $\mathcal{O}_\varepsilon(x)$ chooses a uniformly random $\alpha_x \xleftarrow{\mathrm{R}} [-\varepsilon/2, +\varepsilon/2]$ which we call the "randomizer" of the query x and takes $y = f(x) + \alpha_x$. Then $\mathcal{O}_\varepsilon(x)$ will round y as follows: let $\lfloor y \rfloor_\varepsilon$ denote the the integer multiple of ε that is closest to y. The oracle outputs $\mathcal{O}_\varepsilon(x) = \lfloor y \rfloor_\varepsilon$. Note that it always holds that $\mathcal{O}_\varepsilon \in [f(x) \pm \varepsilon]$, and so by the definition of A and $\mathcal{O}_\varepsilon$, it holds that

$$\Pr_{\mathcal{O}_\varepsilon, A}[A^{\mathcal{O}_\varepsilon}(z) = \Pi(z)] \geq 1 - n^{-\omega(1)} \tag{1}$$

for all $z \in \Pi_Y \cup \Pi_N$ (where $\Pi(z) = 1$ if $z \in \Pi_Y$ and $\Pi(z) = 0$ if $z \in \Pi_N$). The reason is that one can choose and fix the randomness of $\mathcal{O}_\varepsilon$ first and then Inequality 1 holds by the definition of A. In the following, let $p(n) = \operatorname{poly}(n)$ be an upper bound on the number of oracle queries made by A and let ω denote random coins used to run A. Without loss of generality we assume that A does not ask any query x more than once. We also write $\mathcal{O}_\varepsilon(x)$ more explicitly as $\mathcal{O}_\varepsilon(x, \alpha_x)$ to denote the value of the randomizer α_x used for the answer to the query x.

To prove that $\Pi \in \mathbf{AM}$ it suffices to show an $\mathbf{AM}$ protocol where either the verifier rejects or (if the prover is honest) with high probability the verifier does not reject and the output of the verifier is statistically close to the output of $A^{\mathcal{O}_\varepsilon}$. This way the verifier either catches the cheating prover or gets a good emulation of $A^{\mathcal{O}_\varepsilon}$ which she can use to take her final decision and choose to accept or reject.

The intuition. The idea is that the verifier will select random coins ω for the execution of A along with real numbers $\alpha_1, \dots, \alpha_{p(n)}$ drawn uniformly at random from $[\pm\varepsilon/2]$ and send these to the prover, and the prover will use ω to run A responding oracle queries according to $\mathcal{O}_\varepsilon$ while using α_i as the randomizer for the i'th query x_i. The prover sends back all the $f(x_i)$ to the verifier, who checks for all $i \in [p(n)]$ using $f(x_i)$ and α_i that the responses $\lfloor f(x_i) + \alpha_i \rfloor_\varepsilon$ give a consistent and accepting execution of the reduction. Furthermore, we assumed that $f \in \mathbb{R}\text{-}\mathbf{TFAM}$, so let R be the corresponding relation guaranteed by Definition 3.1. Since R is an $\mathbf{AM}$ relation, we will also require that the prover give a proof that all $f(x_i)$'s are correct.

Completeness of this strategy is clear because the honest prover can always prove that $(x, f(x)) \in R$, and so because each oracle query x is answered with the same distribution as $A^{\mathcal{O}_\varepsilon}$. Soundness follows as well. The reason is that the only time the prover can bias the value $\lfloor f(x_i) + \alpha_i \rfloor_\varepsilon$ is when $(f(x_i) + \alpha_i \mod \varepsilon) \approx \varepsilon/2$ for some i, i.e., by slightly perturbing $f(x_i)$ the prover can cause $f(x_i) + \alpha_i$ to be rounded either up or down. But it is easy to see that this bad situation is unlikely, namely for each query, $\Pr_{\alpha_i}[(f(x_i) + \alpha_i \mod \varepsilon) \in [\varepsilon/2 \pm \delta]] \leq O(\delta/\varepsilon)$. By taking $\delta/\varepsilon \ll 1/p(n)$, where $p(n)$ is the total number of queries, the verifier gets an emulation of the reduction which is statistically close to an honest emulation and therefore the soundness of the protocol follows from the definition of the reduction.

The AM protocol for the language Π.

Protocol 3.6. *Common input: instance z.*

1. *The verifier V_A sends a random seed ω that will be used to execute A. V_A also sends random numbers $\alpha_1, \dots, \alpha_{p(n)} \xleftarrow{R} [-\varepsilon/2, +\varepsilon/2]$.*

2. *The prover P_A emulates the execution of A using random coins ω, where the prover answers the i'th oracle query x_i with $\mathcal{O}_\varepsilon(x_i, \alpha_i) = \lfloor f(x_i) + \alpha_i \rceil_\varepsilon$. The prover sends back to the verifier the values $(x_i, f(x_i))$ for all $i \in [p(n)]$.*

3. *In parallel for all $i \in [p(n)]$, the verifier engages the prover in the **AM** protocol that $f(x_i) = y_i$ with approximation error $\frac{\varepsilon}{np(n)}$. If any of these protocols reject, then reject.*

4. *The verifier checks for $i = 1, \ldots, p(n)$ that emulating A with the i'th query x_i answered by $\mathcal{O}_\varepsilon(x_i, \alpha_i) = \lfloor f(x_i) + \alpha_i \rceil_\varepsilon$ leads to A asking the $i+1$'st query x_{i+1}, and so on, and accepts iff A accepts.*

. . . .

Since $\mathbf{BPP}^{\mathbb{R}\text{-}\mathbf{TFAM}}$ is closed under complement, proving that the above protocol decides Π suffices to prove the theorem.

Completeness. For any $z \in \Pi_Y$, the queries x_i are answered with $\mathcal{O}_\varepsilon(x_i, \alpha_i)$. Namely, they are answered according to $\mathcal{O}_\varepsilon$ where the randomizers are chosen by the verifier, and therefore by Inequality 1 the verifier accepts with overwhelming probability.

Soundness. Fix any $z \in \Pi_N$, and let P' be a possibly cheating prover. For each query x_i suppose that P' returns a value $f'(x_i)$ as his claim for $f(x_i)$, and let $y_i' = f'(x_i) + \alpha_i$ and $y_i = f(x_i) + \alpha_i$. If for any $i \in [p(n)]$ it holds that $|y_i - y_i'| > \frac{\varepsilon}{np(n)}$ which is equivalent to $|f(x_i) - f'(x_i)| > \frac{\varepsilon}{np(n)}$, then with overwhelming probability one of the **AM** protocols in Step 3 will fail by the soundness condition of the $\mathbb{R}$-**TFAM** relation, and so V_A will reject.

So let suppose that the strategy of P' is restricted to always claim some $f'(x_i)$ as the value of $f(x_i)$ such that $|f(x_i) - f'(x_i)| \leq \frac{\varepsilon}{np(n)}$ and therefore it holds that $|y_i - y_i'| \leq \frac{\varepsilon}{np(n)}$. Now look at the oracle answer $\lfloor y_1' \rceil_\varepsilon$ used for the first query x_1 in the emulation of the reduction. If it holds that $(y_1 \mod \varepsilon) \notin [\varepsilon/2 \pm \delta]$ for $\delta = \frac{\varepsilon}{np(n)}$, then the value of $\lfloor y_1 \rceil_\varepsilon$ is robust to $\pm\delta$ change in y_1, and so even if the prover cheats in his claim $f'(x_1)$ about $f(x_1)$ by $\pm\delta$, it will still hold that $\lfloor y_1' \rceil_\varepsilon = \lfloor y_1 \rceil_\varepsilon$. But since α_1 was chosen at random from $[\pm\varepsilon/2]$, it is easy to see that:

$$\Pr_{y_1 \xleftarrow{\text{R}} [f(x_1)\pm\varepsilon/2]} [(y_1 \mod \varepsilon) \in [\varepsilon/2 \pm \delta]] = 2\delta/\varepsilon.$$

It means that with probability $\geq 1 - 2\delta/\varepsilon$ over the choice of $\alpha_i \xleftarrow{\text{R}} [\pm\varepsilon/2]$ the prover does not have any control over the first oracle answer used in the emulation of the reduction. The same argument holds for the i'th query for any $i \in [p(n)]$.

It follows by the union bound that the total statistical distance between the set of query/answer pairs generated by $A^{\mathcal{O}_\varepsilon}$ and the query/answer pairs generated by (P', V_A) is bounded by at most $\frac{2}{n}$. Therefore, the probability that (P', V_A) accepts is bounded by at most $\leq 2/n + n^{-\omega(1)}$. Repeating the overall protocol in parallel one can reduce this error to be negligible. $\qquad\square$

Theorem 1.1 together with Proposition 3.5 yield Corollary 1.2, namely $\mathbf{BPP}^{\mathbf{SZK}} \subseteq \mathbf{AM} \cap \mathbf{coAM}$.

4 Reductions that Imply Checkability of SAT

We saw in Section 1.2.1 that a result of Beigel as cited in [6] (Theorem 1.3) along with the fact that **TFNP** is trivially checkable implies the following theorem (see Definition 2.3 for a formal definition of **TFNP**).

Theorem 4.1. *For any relation $R \in$ **TFNP**, if* SAT *can be reduced to (the search problem of) R through a randomized reduction, then* SAT *is checkable.*

4.1 Extending Beigel's theorem to testability

We will prove Theorem 4.2 a variant of Theorem 1.3 for the case in which one of the reductions in the hypothesis has the extra feature of being worst-case to average case (see Definition 2.8). From this stronger assumption we conclude a stronger consequence as follows.

Theorem 4.2 (Restating Theorem 1.5, formally). *Let $\delta = \delta(n)$, let R and R' be relations, and let D be an ensemble of distributions. Suppose that solving R reduces to $(1-\delta)$-solving R' over D, and also suppose that there is a randomized (worst-case to worst-case) reduction from R' to R. Then if R' is δ-testable over D (resp. non-uniformly), then R and R' are checkable (resp. non-uniformly).*

Proof. (of Theorem 4.2) We first prove the theorem when there is no advice: $t(n) = 0$.

Let A_R be the randomized oracle reduction from solving R to $(1-\delta)$-solving R' over D and let $\ell = \text{poly}(n)$ be an upper-bound on the length of the oracle queries of A_R. Notice that, in particular, A_R is also a worst-case to worst-case reduction from R to R'. Let $A_{R'}$ be the randomized (worst-case to worst-case) reduction from R' to R. By Theorem 1.3 and the fact that A_R is also a worst-case to worst-case reduction from R to R', it follows that R is checkable if and only if R' is checkable. Therefore in the following it suffices to show that R is checkable.

Now suppose that $T_{R'}$ is a δ-tester for R'. Given input x, the checker $C_R^{\mathcal{O}}(x)$ does the following.

- For all $i \leq \ell(n)$, C_R runs the δ-tester $T_{R'}$ over the oracle $A_{R'}^{\mathcal{O}}$ which is supposed to decide R' with probability at least $1 - \delta$ over D_ℓ for the input length i. C_R rejects if $T_{R'}$ rejects.

- $C_R(x)$ runs the reduction A_R on input x. For each query y that A_R makes, C_R runs $A_{R'}^{\mathcal{O}}(y)$ to find the answer. C_R outputs whatever A_R outputs.

The completeness of C_R is immediate, because if $\mathcal{O}$ is a perfect oracle for R' then for each query y, $A_{R'}^{\mathcal{O}}(y) \in R'(y)$ holds with overwhelming probability. In this case with overwhelming probability $T_{R'}$ will accept also and C_R will output $z \in R(x)$.

The soundness of C_R holds by the soundness of $T_{R'}$ and the definition of A_R. Namely either $A_{R'}^{\mathcal{O}}$ $(1-\delta)$-solves R' over D_i for $i \leq \ell(n)$, or else $T_{R'}$ rejects with overwhelming probability. In the first case, by the correctness of A_R, the checker C_R will find a correct output $\in R(x)$ with overwhelming probability.

For non-uniform testers, it is clear that the above reduction still holds except one needs to hardwire into C_R the advice strings for all input lengths $i \leq \ell(n)$ that $A_{R'}^{\mathcal{O}}$ might query. $\square$

Theorem 4.3 (Formal statement of Theorem 1.7). *Let R be any **NP** relation, $\delta = 1/\text{poly}(n)$, and D_n be any ensemble of samplable distributions. Then R can be δ-tested over D_n given $\lceil \log \frac{2}{\delta} \rceil = O(\log n)$ bits of advice. In particular the advice for length n inputs can be any s_n such that $s_n \leq \Pr_{y \leftarrow D_n}[R(y) \neq \varnothing] \leq s_n + \delta/2$.*

The intuition. Suppose that we are given $s_n \approx \Pr_{y \leftarrow D_n}[R(y) \neq \varnothing]$ (the approximate fraction of YES instances of R over D_n). If we sample enough number of points $y_1, \ldots, y_k \leftarrow D_n$, then by Chernoff we anticipate $\approx s_n$ fraction of y_i's to satisfy $R(y_i) \neq \varnothing$. As the first step the checker for R simply verifies that $s_n \approx |\{i \mid \mathcal{O}(y_i) \neq \varnothing\}|/k$ holds. This yet does not mean that with high probability over $y \leftarrow D_n$, $\mathcal{O}(y)$ returns the right answer. But since R is an **NP**-relation we can always make sure that if $\mathcal{O}(y)$ returns $z \neq \varnothing$, then $(y, z) \in R$. By enforcing this extra check over the solutions $z_i = \mathcal{O}(y_i)$, the oracle $\mathcal{O}$ can be wrong only in "one direction": to return $\varnothing$ for a query y that $R(y) \neq \varnothing$. But if $\mathcal{O}$ does so significantly, then it will change its bias $\Pr_{y \leftarrow D_n}[\mathcal{O}(y) \neq \varnothing] \ll s_n$ and it can be detected. A very similar trick is used in [12, 9, 4]. The difference between our setting and [12, 9, 4] is that they deal with *provers* (rather than oracles) that are stateful and might cheat more intelligently by answering their queries depending on all the queries that they are asked. That makes the job of [12, 9, 4] potentially harder, but they bypass this difficulty by putting strong restrictions on the adaptivity of the reduction (which is not the case for our result). Also [29, 33] uses this technique in another setting where the advice $s_n \approx \Pr_{y \leftarrow D_n}[\mathcal{O}(y) \neq \varnothing]$ is used to construct a non-uniform reduction that $(1 - 1/\operatorname{poly}(n))$-solves R over D_n given access to any oracle that solves R correctly only with probability $\geq 1/2 + n^{-1/3+\varepsilon}$.

Proof. (of Theorem 4.3) Let $\mathcal{O}$ be the oracle that is going to be tested for the relation R. The checker algorithm $A_R^{\mathcal{O}}$ acts as follows.

1. Let $k = n/\delta^2$. For $i \in [k]$ sample $x_i \xleftarrow{\text{R}} \{0,1\}^n$ uniformly.

2. For $i \in [k]$, ask x_i from $\mathcal{O}$ to get $y_i = \mathcal{O}(x_i)$.

3. If for any $i \in [k]$, it holds that $y_i \neq \varnothing$ but $(x_i, y_i) \notin R$ (which can be tested efficiently), then halt and output $\perp$.

4. If $\frac{|\{i \colon \mathcal{O}(x_i) \in R(x)\}|}{k} < s - \frac{\delta}{6}$, then halt and output $\perp$, otherwise accept and output 1.

Completeness. If $\mathcal{O}(x) \in R(x)$ for all $x \in \{0,1\}^n$, then A_R never halts in Step 3. We show that the probability of halting in Step 4 is negligible. Let $p = \Pr_{x \leftarrow D_n}[R(x) \neq \varnothing]$ and $p' = \frac{|\{i \colon \mathcal{O}(x_i) \in \mathbb{R}(x)\}|}{k}$ be the empirical estimate of p. By Chernoff, it holds that $\Pr_{x_1, \ldots, x_k}[|p' - p| > \varepsilon] < 2e^{-2k\varepsilon^2}$. Now since it is guaranteed that $s \leq p$, therefore it holds that

$$\Pr[p' < s - \delta/6] \leq \Pr[p' < p - \delta/6] \leq \Pr[|p' - p| > \delta/6] < 2e^{-2k(\delta/6)^2} = 2e^{-n^2/18} = \operatorname{neg}(n).$$

Soundness. Suppose $\Pr_{x \leftarrow D_n}[\mathcal{O}(x) \notin R(x)] \geq \delta$. Then either it holds that $\Pr_{x \leftarrow D_n}[R(x) = \varnothing \wedge \mathcal{O}(x) \neq \varnothing] \geq \delta/6$ or it holds that $\Pr_{x \leftarrow D_n}[R(x) \neq \varnothing \wedge \mathcal{O}(x) \notin R(x)] \geq 5\delta/6$. We show that in both cases A_R halts with overwhelming probability.

If $\Pr_{x \leftarrow D_n}[R(x) = \varnothing \wedge \mathcal{O}(x) \neq \varnothing] \geq \delta/6$ then with probability at least $1 - (1 - \delta/6)^k = 1 - \operatorname{neg}(n)$ one of x_i's is sampled such that $R(x_i) = \varnothing$ and $\mathcal{O}(x_i) \neq \varnothing$. In this case $(x_i, \mathcal{O}(x_i)) \notin R$ and so A_R rejects in Step 3.

On the other hand if $\Pr_{x \leftarrow D_n}[R(x) \neq \varnothing \wedge \mathcal{O}(x) \notin R(x)] \geq 5\delta/6$, because $p \leq s + \delta/2$ then

$$\Pr_{x \leftarrow D_n}[\mathcal{O}(x) \in R(x)] \leq \Pr_{x \leftarrow D_n}[R(x) \neq \varnothing \wedge \mathcal{O}(x) \in R(x)] \leq \Pr_{x \leftarrow D_n}[R(x) \neq \varnothing] - 5\delta/6$$
$$= p - 5\delta/6 \leq s + \delta/2 - 5\delta/6 = s - \delta/3.$$

13

So for each i, $\Pr_{x_i \leftarrow D_n}[\mathcal{O}(x_i) \in R(x_i)] \le s - \delta/3$, and thus by Chernoff it holds that

$$\Pr[p' \ge s - \delta/6] = \Pr[p' \ge (s - \delta/3) + \delta/6] < 2e^{-2k(\delta/6)^2} = 2e^{-n^2/18} = \mathrm{neg}(n).$$

But if $p' < s - \delta/6$, then A_R halts in Step 4.

$\square$

Worst-case to average-case reductions for NP

For definitions of worst-case to average-case reductions, see Definition 2.8.

Corollary 4.4 (Formal statement of Corollary 1.6). *Let R be an* **NP***-relation, let $\delta = 1/\mathrm{poly}(n)$ and let D be any efficiently samplable ensemble of distributions. If* SAT *reduces to $(1 - \delta)$-solving R over D, then* SAT *has a non-uniform checker.*

Corollary 4.5 (Formal statement of Corollary 1.8). *Let $\delta = 1/\mathrm{poly}(n)$, and let f be an efficiently computable family of functions. If* SAT *reduces to $(1 - \delta)$-inverting f, then* SAT *is* uniformly checkable.

Proof. (of Corollaries 4.4 and 4.5) Since SAT is **NP** complete, there is a (worst-case to worst-case) randomized reduction from the **NP** relation R of Theorem 4.3 to the relation SAT. Therefore Corollary 4.4 follows from Theorem 4.2 and Theorem 4.3 immediately.

Corollary 4.5 also follows from Theorem 4.2 and Theorem 4.3 similarly, but this time we do not need the advice because of the following. Even though R^f might not be a total relation, but it still holds that

$$\Pr_{y \leftarrow D_i^f}[R^f(y) \ne \varnothing] = \Pr_{y \leftarrow f(U_i)}[y \text{ is invertible}] = 1.$$

Thus here we can use the constants $s_i = 1$ as the advice for all the query lengths $i \le \mathrm{poly}(n)$ that reduction might ask and apply Theorem 4.3, eliminating the need for advice. $\square$

5   Acknowledgements

References

[1] W. Aiello and J. Håstad. Statistical zero-knowledge languages can be recognized in two rounds. *Journal of Computer and System Sciences*, 42(3):327–345, 1991. Preliminary version in *FOCS'87*.

[2] W. Aiello and J. Håstad. Statistical zero-knowledge languages can be recognized in two rounds. *Journal of Computer and System Sciences*, 42(3):327–345, 1991. Preliminary version in *FOCS'87*.

[3] M. Ajtai. The worst-case behavior of schnorr's algorithm approximating the shortest nonzero vector in a lattice. In *STOC '03*, pages 396–406, New York, NY, USA, 2003. ACM. ISBN 1-58113-674-9. doi: http://doi.acm.org/10.1145/780542.780602.

[4] A. Akavia, O. Goldreich, S. Goldwasser, and D. Moshkovitz. On basing one-way functions on np-hardness. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing (STOC)*, pages 701–710, 2006.

[5] L. Babai and S. Laplante. Stronger separations for random-self-reducibility, rounds, and advice. In *In IEEE Conference on Computational Complexity*, pages 98–104, 1999.

[6] M. Blum and S. Kannan. Designing programs that check their work. *J. ACM*, 42(1):269–291, 1995. ISSN 0004-5411. doi: http://doi.acm.org/10.1145/200836.200880.

[7] M. Blum and S. Micali. How to generate cryptographically strong sequences of pseudo-random bits. *SIAM J. Comput.*, 13(4):850–864, 1984. ISSN 0097-5397. doi: http://dx.doi.org/10.1137/0213053.

[8] M. Blum, M. Luby, and R. Rubinfeld. Self-testing/correcting with applications to numerical problems. *Journal of Computer and System Sciences*, 47(3):549–595, 1993.

[9] A. Bogdanov and L. Trevisan. On worst-case to average-case reductions for np problems. *SIAM Journal on Computing*, 36(4):1119–1159, 2006.

[10] G. Brassard. Relativized cryptography. In *Proceedings of the 20th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 383–391. IEEE Computer Society, 1979.

[11] S. Even, A. L. Selman, and Y. Yacobi. The complexity of promise problems with applications to public-key cryptography. *Inf. Control*, 61(2):159–173, 1984. ISSN 0019-9958. doi: http://dx.doi.org/10.1016/S0019-9958(84)80056-X.

[12] J. Feigenbaum and L. Fortnow. Random-self-reducibility of complete sets. *SIAM Journal on Computing*, 22(5):994–1005, 1993.

[13] J. Feigenbaum, L. Fortnow, C. Lund, and D. Spielman. The power of adaptiveness and additional queries in random-self-reductions. *Computational Complexity*, 4:338–346, 1994.

[14] L. Fortnow. The complexity of perfect zero-knowledge. *Advances in Computing Research: Randomness and Computation*, 5:327–343, 1989.

[15] L. Fortnow, J. Rompel, and M. Sipser. On the power of multi-prover interactive protocols. *Theoretical Computer Science*, 134(2):545–557, 1994.

[16] O. Goldreich. On promise problems (a survey in memory of Shimon Even [1935-2004]). Technical Report TR05–018, Electronic Colloquium on Computational Complexity, February 2005.

[17] O. Goldreich and S. Vadhan. Comparing entropies in statistical zero-knowledge with applications to the structure of SZK. Technical Report TR98-063, Electronic Colloquium on Computational Complexity, 1998.

[18] O. Goldreich and S. P. Vadhan. Comparing entropies in statistical zero knowledge with applications to the structure of SZK. In *IEEE Conference on Computational Complexity*, pages 54–73. IEEE Computer Society, 1999.

[19] O. Goldreich, S. Goldwasser, and D. Ron. Property testing and its connection to learning and approximation. *J. ACM*, 45(4):653–750, 1998. ISSN 0004-5411. doi: http://doi.acm.org/10.1145/285055.285060.

[20] I. Haitner, M. Mahmoody-Ghidary, and D. Xiao. A new sampling protocol with applications to basing cryptographic primitives on the hardness of **NP**, 2009.

[21] I. Haitner, A. Rosen, and R. Shaltiel. On the (im)possibility of arthur-merlin witness hiding protocols. In *Theory of Cryptography, Fourth Theory of Cryptography Conference, TCC 2009*, 2009.

[22] J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999. Preliminary versions in *STOC'89* and *STOC'90*.

[23] E. Hemaspaandra, A. V. Naik, M. Ogihara, and A. L. Selman. P-selective sets and reducing search to decision vs. self-reducibility. *J. Comput. Syst. Sci.*, 53(2):194–209, 1996. ISSN 0022-0000. doi: http://dx.doi.org/10.1006/jcss.1996.0061.

[24] T. Holenstein. Private communication. 2009.

[25] R. Impagliazzo and A. Wigderson. **P** = **BPP** if **E** requires exponential circuits: Derandomizing the XOR lemma. In *Proc. 29th STOC*, pages 220–229. ACM, 1997.

[26] N. Megiddo and C. H. Papadimitriou. On total functions, existence theorems and computational complexity. *Theor. Comput. Sci.*, 81(2):317–324, 1991. ISSN 0304-3975. doi: http://dx.doi.org/10.1016/0304-3975(91)90200-L.

[27] D. Micciancio and O. Regev. Worst-case to average-case reductions based on gaussian measures. In *FOCS '04*, pages 372–381, Washington, DC, USA, 2004. IEEE Computer Society. ISBN 0-7695-2228-9. doi: http://dx.doi.org/10.1109/FOCS.2004.72.

[28] N. Nisan and A. Wigderson. Hardness vs randomness. *Journal of Computer and System Sciences*, 49(2):149–167, Oct. 1994. Preliminary version in FOCS' 88.

[29] R. O'Donnell. Hardness amplification within np. *J. Comput. Syst. Sci.*, 69(1):68–94, 2004. ISSN 0022-0000. doi: http://dx.doi.org/10.1016/j.jcss.2004.01.001.

[30] C. H. Papadimitriou. On the complexity of the parity argument and other inefficient proofs of existence. *J. Comput. Syst. Sci.*, 48(3):498–532, 1994. ISSN 0022-0000. doi: http://dx.doi.org/10.1016/S0022-0000(05)80063-7.

[31] R. Pass. Parallel repetition of zero-knowledge proofs and the possibility of basing cryptography on np-hardness. In *IEEE Conference on Computational Complexity*, pages 96–110, 2006.

[32] A. Sahai and S. Vadhan. A complete problem for statistical zero knowledge. *Journal of the ACM*, 50(2):196–249, 2003. Preliminary version in *FOCS'97*.

[33] Trevisan. On uniform amplification of hardness in NP. In *STOC: ACM Symposium on Theory of Computing (STOC)*, 2005.

[34] S. P. Vadhan. *A Study of Statistical Zero-Knowledge Proofs*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 1999.

[35] A. C. Yao. Protocols for secure computations. pages 160–164, 1982.

A   Omitted Proofs

AM protocol for entropy

Proof. We give a simple proof for completeness. Recall the following facts. For two circuits Y_1 and Y_2 sampling distributions, let their concatenation $Y_1 Y_2$ be the joint circuit sampling the product distribution. For all $\varepsilon > 0$, it is possible given any $y > 0$ to construct in $\text{poly}(n, 1/\varepsilon)$ time a circuit Z_y such that $|H(Z_y) - y| < \varepsilon/100$.

It is known that ED can be reformulated so that the gap between X_1 and X_2 is $\varepsilon/2$ rather than 1, and still the problem remains complete for **SZK**, so let us assume this. Since $\text{ED} \in \textbf{SZK} \subseteq \textbf{AM}$ [2, 17], therefore given input (X, y), the verifier can run in parallel the **AM** protocol for ED on inputs $(X Z_1, Z_y)$ and (Z_{y+1}, X) and accept iff both executions accept.   □

Proof of Proposition 3.5. We show how to implement an ED oracle using a $\mathbb{R}$-**TFAM** oracle for any $\varepsilon \leq 1/2$. Consider the function $f(X_1, X_2) = H(X_1) - H(X_2)$. This function is in $\mathbb{R}$-**TFAM** because by Lemma 3.4 the entropy of a circuit can be approximated using an **AM** protocol.

We claim that given any oracle $\mathcal{O}$ that solves (the search problem of) the relation R with up to ε error, one can decide ED: given (X_1, X_2), query (X_1, X_2) from $\mathcal{O}$ to get $y = \mathcal{O}(X_1, X_2)$ and accept iff $y > 0$. Since we are guaranteed that $|(H(X_1) - H(X_2)) - y| \leq \varepsilon$, it therefore holds that if $H(X_1) > H(X_2) + 1$ then $y > 1 - \varepsilon$ and if $H(X_1) < H(X_2) - 1$ then $y < -1 + \varepsilon$.   □

Proof sketch of Theorem 1.3

Let C_Π be a checker for Π. In order to check Π', given the input x, the checker algorithm $C_{\Pi'}$ first runs the reduction $A'(x)$ and for any query like y that $A'(x)$ wants to ask from its R oracle, $C_{\Pi'}$ does the following. $C_{\Pi'}$ runs the checker algorithm $C_\Pi(y)$ over the "oracle" $A^{\mathcal{O}}$. If it leads to reject, C_Π rejects as well, but if it did not reject and returned the output z, $C_{\Pi'}$ safely uses the answer z for the query y and continues running the reduction $A'(x)$. It is easy to see C_Π's completeness and soundness of follow from those of $C_{\Pi'}$ and the definition of reductions A and A'.

Proof sketch of $\mathbf{P^{SZK}} \subseteq \mathbf{AM} \cap \mathbf{coAM}$

Vadhan (as cited by Goldreich [16]) proves the following:

Theorem A.1 (Vadhan, appears as Theorem 5.4 in [16]). *Let $\Pi = (\Pi_Y, \Pi_N)$ be such that there exist sets S_Y, S_N satisfying $S_Y \cup S_N = \{0,1\}^*$, $\Pi_Y \subseteq S_Y$, $\Pi_N \subseteq S_N$, $(S_Y, \Pi_N) \in \mathbf{NP}$, $(S_N, \Pi_Y) \in \mathbf{NP}$. Then it holds that $\mathbf{P}^\Pi \subseteq \mathbf{NP} \cap \mathbf{co\text{-}NP}$.*

This result can be interpreted the following way. Consider problems Π that "extend" to a **TFNP** search problem in the following way: there exist disjoint S_Y, S_N containing Π_Y, Π_N respectively such that the relation

$$(x, bw) \in R \quad \Leftrightarrow \quad (b = 1 \wedge (x, w) \in S_Y) \vee (b = 0 \wedge (x, w) \in S_N)$$

is in **TFNP**. This is equivalent to the hypothesis of Theorem A.1, and solving R immediately implies solving Π. Now combined with the fact that $\mathbf{P^{TFNP}} \subseteq \mathbf{NP} \cap \mathbf{co\text{-}NP}$ one obtains Vadhan's result.

This definition naturally generalizes to problems Π that "extend" to **TFAM** problems. As we argued in Proposition 3.5, the **SZK**-complete problem entropy difference can be extended to the **TFAM** problem of computing $f(X_1, X_2) = H(X_1) - H(X_2)$. For the same reason that $\mathbf{P^{TFNP}} \subseteq \mathbf{NP} \cap \mathbf{co\text{-}NP}$, it also holds that $\mathbf{P^{TFAM}} \subseteq \mathbf{AM} \cap \mathbf{coAM}$. Therefore, one can observe that

$$\mathbf{P^{SZK}} = \mathbf{P^{ED}} \subseteq \mathbf{P^{TFAM}} \subseteq \mathbf{AM} \cap \mathbf{coAM}$$