On Worst-Case to Average-Case Reductions for NP Problems

Andrej Bogdanov*　　　Luca Trevisan†

July 23, 2003

Abstract

We show that if an NP-complete problem has a non-adaptive self-corrector with respect to a samplable distribution then coNP is contained in AM/poly and the polynomial hierarchy collapses to the third level. Feigenbaum and Fortnow show the same conclusion from the stronger assumption that an NP-complete problem has a non-adaptive random self-reduction.

Our result shows that it is impossible (using non-adaptive reductions) to base the average-case hardness of a problem in NP or the security of a one-way function on the worst-case complexity of an NP-complete problem (unless the polynomial hierarchy collapses).

*Computer Science Division, U.C. Berkeley. `adib@cs.berkeley.edu`.

†Computer Science Division, U.C. Berkeley. `luca@cs.berkeley.edu`. Research supported by a Sloan Research Fellowship and an Okawa Foundation Grant.

1 Introduction

Worst-Case versus Average-Case Complexity

A problem in distributional NP [Lev86] is a pair (L, D) where L is an NP decision problem and D is a samplable distribution of instances.[1]

The fundamental question in the study of average-case complexity is whether there are intractable problems in distributional NP. Of course the question can be formalized in different ways depending on how we define intractability (and tractability). For the sake of this paper we will consider a problem (L, D) tractable if for every polynomial $p()$ there is a polynomial time algorithm A such that, for any n, there is a probability at most $1/p(n)$, according to D, to generate an instance of length n on which A makes a mistake, and we say that (L, D) is intractable otherwise.[2]

Results by Ajtai [Ajt96] suggests that the question of whether every distributional NP is tractable may be reduced to the standard question of whether NP $\subseteq$ BPP. Ajtai shows that an algorithm that solves well on average the shortest vector problem (an NP problem) under a certain samplable distribution of instances implies an algorithm that solves, in the worst case, an approximate version of the shortest vector problem, which can be seen as an NP promise problem. If the latter problem were NP-complete, then we would have a reduction relating the average-case hardness of an NP distributional problem to the worst-case hardness of an NP-complete problem. Unfortunately, the latter problem is known to be in NP $\cap$ *co*NP, and therefore it is unlikely to be NP-hard. However, it is conceivable that improved versions of Ajtai's argument could show the equivalence between the average-case complexity of a distributional NP problem and the worst-case complexity of an NP problem.

Ajtai's approach has been extended by Ajtai and Dwork [AD97] and Regev [Reg03], who present public-key cryptosystems whose security (which is a stronger condition than the existence of hard-on-average problems in NP) is equivalent to the worst-case complexity of certain NP promise problems.

Such results re-openened the old question (already present in [DH76, Section 6]) of whether there are cryptosystems that are NP-hard to break, that is, whose security can be based on the assumption that NP $\not\subseteq$ BPP.

[1] Actually, we think of D as being a samplable *ensamble* of distributions, that is, for every n there is a distribution D_n on instances of length n, and there is a polynomial time sampler that on input 1^n samples from D_n. This is the standard convention in cryptography and in the study of "amplification of hardness" and worst-case to average-case reductions for various problems, including problems in NP [O'D02]. It is, however, different from the convention used by Levin [Lev86], which refers to a single distribution over all possible inputs. Impagliazzo [Imp95] shows that the two conventions are, essentially, interchangeable.

[2] This is essentially the definition of *Heuristic Polynomial Time* given by Impagliazzo [Imp95]. One gets, essentially, Levin's definition of Average Polynomial Time [Lev86] by requiring A to output either the right answer or FAIL on every input, and that the probability of answering FAIL when given an instance sampled from D is at most $1/p(n)$. As we explain in the conclusions, our result hold also with respect to Levin's definition of tractability and intractability.

1.1 Previous Work on Worst-case versus Average-case Complexity in NP

As discussed in [Imp95], we know oracles relative to which NP $\not\subseteq$ P/poly but for every NP problem and for every samplable distribution there is a polynomial time algorithm that works well on average. Therefore, any proof that, say, "NP $\not\subseteq$ BPP implies the existence of hard-on-average problems in NP" must use a non-relativizing argument.

Since Ajtai's arguments exploit properties specific problems, it does not seem that relativization results apply to them.

Feigenbaum and Fortnow [FF93] consider the notion of *locally random reduction*, which is a natural way to prove that the average-case complexity of a given problem relates to the worst-case complexity of another one. A locally random reduction from a language L to a language L' with respect to a distribution D is a polynomial-time oracle procedure R such that $R^{L'}$ solves R and, furthermore, each oracle query of $R^{L'}(x)$ is distributed according to D (or, rather, according to D_m where m depends only on the input length of x). Clearly, such a reduction converts a heuristic polynomial time algorithm for L' with respect to D (with sufficiently small error probability) into a BPP algorithm for L. If we could have a locally random reduction from, say, 3SAT to some problem L' in NP, then we would have proved that if NP $\not\subseteq$ BPP then distributional NP contains intractable problems.

Feigembaum and Fortnow show that if there is a *non-adaptive* locally random reduction from a problem L to a problem L' in NP, then L is in *co*NP/poly. In particular, if L is NP-complete, then NP $\subseteq$ *co*NP/poly and the polynomial hierarchy collapses.

Locally random reductions are a natural notion, and they have been used to establish the worst-case to average-case equivalence of PSPACE-complete and EXP-complete.

Previous Results on Crpytography versus NP-hardness

Brassard [Bra79] considers the question of whether there can be a *public key* cryptosystem whose security can be reduced to solving an NP-complete problem. Brassard argues that, under some assumptions on the key-generation algorithm and the encryption procedure, the problem of inverting the encryption is in NP $\cap$ *co*NP, and therefore unlikely to be equivalent to an NP-complete problem. Goldreich and Goldwasser [GG98] revisited the issue more recently, and tried to remove some of the assumptions in Brassard's result, and they showed that the existence of a reduction from an NP-complete problem to the problem of breaking a public-key cryptosystem would imply the collapse of the polynomial hierarchy (under some assumptions on the way the reduction works and/or on the key generation algorithm).

The results of Brassard [Bra79] and of Goldreich and Goldwasser [GG98] refer to the complexity of breaking a cryptosystem for every message and for every key. Clearly, if even such a strong form of attack cannot be NP-hard (under certain assumptions) then neither can the weaker form of attack considered in standard definitions of security (in which the attacker only needs to distinguish the encryptions of two possible messages with noticeable probability). In the setting of *private* key encryption, however, this approach does not seem to work, and it seems necessary to specifically address the issue of the average-case complexity of attacking a construction. In particular, the possibility of private-key encryption is equivalent to the existence of one-way functions, and it is well known that there are "one-way functions" that are NP-hard to invert on all inputs.[3]

A generic reduction from an NP-complete problem to the problem of inverting a one-way function f would be an oracle procedure R such that for some polynomial p and for every oracle A that inverts f on a $1 - 1/p(n)$ inputs of length n, we have that R^A is a BPP algorithm for 3SAT. The techniques of Feigembaum and Fortnow imply that if R is non-adaptive, and if all of its oracle queries are done according to the same distribution (that depends only on the length of the input), then the existence of such a reduction implies that the polynomial hierarchy collapses.

As we explain below, our results show the same conclusion without the assumption on the distribution of the queries made by R^A. (But we still need the assumption that the queries are non-adaptive.)

Our Result

We say that a language L has a worst-case to average-case reduction with parameter δ to a distributional problem (L', D) if there is a reduction R (say, realized by a probabilistic polynomial time algorithm) such that, for every oracle A that agrees with L' on inputs of probability mass $1 - \delta$ according to D on each input length, R^A solves L on every input.

If L and L' are the same language, then the reduction is called a self-corrector, a notion independently introduced by Blum and others [BLR93] and by Lipton [Lip89] in the context of program checking [Blu88, BK95].

Clearly a locally random reduction is also a worst-case to average-case reduction and a random self-reduction is also a self-corrector, but the reverse need not be true.

In this paper we show that if there is a worst-case to average-case reduction with parameter $1/\text{poly}(n)$ from an NP-complete problem L to a distributional NP problem (L, D), then NP $\subseteq$ *co*NP/poly and the polynomial hierarchy collapses.

In particular, if an NP-complete problem has a self-corrector with respect to a samplable distribution, then the polynomial hierarchy collapses.

We first prove the result for the special case in which the distribution D is uniform.

Using a reductions by Impagliazzo and Levin [IL90] and by Ben-David and others [BCGL89], we show that the same is true even the reduction assumes a good-on-average algorithm for the *search* version of L', and even if we measure average-case complexity for L' with respect to an arbitrary samplable distribution D.

The generalization to arbitrary samplable distributions and to search problems also implies that there cannot be any non-adaptive reduction from an NP-complete problem to the problem of inverting a one way function.

[3]For example, take the function that on input a 3SAT formula ϕ and an assignment a, outputs $0.\phi$ if the formula is not satisfied by a, and outputs $1.\phi$ otherwise.

Our result also rules out non-adaptive reductions from an NP-complete problem to the problem of breaking a public-key cryptosystem. The constraint of non-adaptivity of the reduction is incomparable to the constraints in the results of Goldreich and Goldwasser [GG98]/

Comparison with the [FF93]

A locally random reduction R that makes q queries is also a worst-case to average-case reduction with parameter $1/O(q)$. Indeed, if A is an oracle that has agreement, say, $1-1/4q$ with L', and we access the oracle via q queries, each uniformly distributed, there is a probability at least $3/4$ that queries made to A are answered in the same way as queries made to L'.

On the other hand, the restriction that all queries must have the same distribution is quite strong, and one could imagine reductions where the distribution of the queries is somewhat dependent on the input, as long as, for every small subset of possible queries, a majority of the queries land outside of the subset with high probability. Furthermore, locally random reductions lack nice closure properties. We would like to say that if solving L in the worst case is reducible to solving L' on a $1-\delta$ fraction of inputs, and if solving L' on a $1-\delta$ fraction of inputs is reducible to solving L'' on a $1-\delta'$ fraction of inputs, then solving L in the worst case is reducible to solving L'' on a $1-\delta'$ fraction of inputs. Reductions among distributional problems [Lev86] typically produce instances for the target problem that are not necessarily uniformly distributed, but just have a distribution that is *dominated* by the uniform distribution.[4]

For this reason, it is interesting, for starters, to have a generalization of the result of Feigenbaum and Fortnow to the case of a reduction R such that $R^{L'}$ computes L, and each oracle query is made with a probability at most polynomially larger than in the uniform distribution. We could call such reductions *smooth random reductions*. However it seems more interesting to just drop all restrictions on the distribution of the queries of R, and just impose the condition that we are interested in: that R works when given any oracle that solves L' well on average.

Readers who are familiar with the following notions may have noted that the relation between locally random reductions and our notion of worst-case to average-case reduction is similar to the relation between one-round private information retrieval and locally checkable codes. In one-round private information retrieval, a user is given oracle access to the encoding of a certain string, and wants to retrieve one bit of the string by making a bounded number of queries; the restriction is that the i-th query must have a distribution independent of the bit that one is interested in. In a locally checkable code, a decoder is given oracle access to the encoding of a certain string, and the encoding has been corrupted in a δ fraction of places; the decoder wants to retrieve a bit of the original string by making a bounded number of queries. The notion of a smooth code, which is the analogue of a smooth random reduction, has also been studied. In a smooth code, a decoder is given oracle access to the encoding of a certain string, and wants to retrieve one bit of the string by making a bounded number of queries; the restriction is that the distribution of each query should be dominated by the uniform distribution.

For unbounded users/decoder the three notions have been shown equivalent [KT00, GKST02], but the same methods do not work in the computationally bounded setting studied in this paper. One

[4] That is, no instance is produced with a probability more than polynomially larger than in the uniform distribution.

step in our proof is, however, inspired by the techniques used to show this equivalence.

Our Proof

As in the work of Feigenbaum and Fortnow, we use the fact that problems in coAM/poly cannot be NP-complete unless the polynomial hierarchy collapses. So our goal is to show that if L is in NP and it has a $1/\text{poly}(n)$ worst-case to average-case reduction to a language L' in NP, then L is also in coAM/poly.

We start by discussing the case in which D is the uniform distribution.

The Feigenbaum-Fortnow protocol. Let us first briefly review the proof of Feigenbaum and Fortnow. Given x, a prover wants to prove that $R^{L'}(x)$ rejects, where R makes q non-adaptive queries, each uniformly distributed. The (non-uniform) verifier generates k independent computations of $R^{L'}(x)$ and sends to the prover all the kq queries generated in all the k runs. The prover has to provide all the answers, and certificates for all the YES answers. The verifier, non-uniformly, knows the overall fraction p of queries of $R^{L'}(x)$ whose answer is YES and, if k is large enough, the verifier expects the number of YES answers from the prover to be concentrated around kqp, and it rejects if the prover gives fewer than $kqp - O(\sqrt{kq})$ YES answers. A cheating prover can only cheat by sayng NO on a YES instance, and cannot do so on more than $O(\sqrt{kq})$. If k is sufficiently larger than q, then with high probability either the verifier rejects or at least one of the k computations of $R^{L'}(x)$ yields correct answers.

Handling Smooth Reductions. Notice that the Feigenbaum-Fortnow protocol can be used with every oracle procedure $R^{L'}(x)$, provided that given x we can get a good estimate of the average number of oracle queries of $R^{L'}(x)$ that are answered YES. Suppose that R is a smooth random reduction, that is, each possible query is generated with probability at most polynomially larger than in the uniform distribution. We devise a *hiding protocol* in which the verifier either rejects or gets a good estimate of the fraction of queries of $R^{L'}(x)$ that are answered YES. We pick at random a query y of $R^{()}(x)$, that is, we select randomness for $R^{()}(x)$, we get q queries, and pick at random one of them. Then we "immerse" y in a random position in a sequence of k random elements of L of the same length, and we give the sequence to the prover. The prover has to say which of the elements in the sequence is a YES instance, and give a certificate for each of them; it is also required to give at least $pk - O(\sqrt{k})$ certificates, where p is the fraction of elements of L of that length that are YES instances (the verifier is given p non-uniformly). A cheating prover can give at most $O(\sqrt{k})$ wrong answers and, roughly speaking, if k is large enough, more than $\sqrt{k}$ elements of the sequence look like queries of $R^{()}(x)$. With high probability, either the verifier rejects or it gets the right answer for y. Repeating the process in parallel many times gives a good estimate of the fraction of queries that are answered YES. This argument is already powerful enough to generalize the result of Feigenbaum and Fortnow to smooth reductions.

Handling General Reductions. Let R be an arbitrary δ worst-case to average-case reduction from L to L' such that $R^{()}(x)$ makes q queries of length m. Intuitively, we would like to convert

R to a smooth reduction as follows: fix a threshold $t = q/\delta$, then for every query made by $R^{()}(x)$ compute the probability that that query be generated by $R^{()}(x)$. Call a possible query "heavy" if it is generated with probability more than $t/2^m$ be the reduction, and "light" otherwise. Ask light queries to the oracle, and do not ask the heavy ones, but proceed as if the heavy ones had been answered NO. Let R' be this modified procedure. Then $R'^{L'}(x)$ behaves like $R^A(x)$ where A differs from L' only on the queries that have a probability more than $t/2^m$ of being generated, so that A agrees with L' on at least a $1-\delta$ fraction of inputs, and $R'^{L'}(x)$ works with high probability. Furthermore, R' is smooth by construction.[5] The problem is that it is hard to compute, or even to prove in an AM protocol, the exact value of the probability that a given query be asked by R. We will settle for approximations, and, roughly speaking, R' will ask a query y to the oracle if the probability of y is less than $t(1-\epsilon)/2^m$ and will simulate a NO answer to y if the probability of y is more than $t(1+\epsilon)/2^m$, and we get in trouble when the probability is in the middle. By picking t at random in a certain range, instead of fixing it to δ/q, we can make sure that with high probability there are few queries for which we get in trouble. Given a query y and a threshold t, the Goldwasser-Sipser [GS86] protocol can be used to prove that the y is "approximately heavy."[6] Unfortunately there is no good protocol to prove that y is an approximately light query. Instead, we show how to use the Aiello-Håstad [AH91] protocol to convince the verifier that the fraction of light queries is approximately some value ℓ. Then the verifier runs a modified immersion protocol to estimate the fraction of heavy queries that are answered YES. In the modified immersion protocol, the prover receives a random query of $R^{()}(x)$ immersed in a sequence or uniformly random strings. The prover has to provide a certificate for each YES instance, and also, for each string that is a heavy query of the protocol, a proof that it is a heavy query. Then the verifier can check that the fraction of queries identified as heavy is about $1-\ell$, and get a good estimate r of the fraction of heavy queries whose answer is YES. Finally, we run a modified Feigenbaum-Fortnow protocol in which we give to the prover the queries of k instantiations of $R^{()}(x)$. The prover has to provide certificates for all the YES instances, and proofs of heaviness for all the heavy queries. The verifier checks that about $(1-\ell)kq$ queries are claimed to be heavy, a fraction r of them has certificates, and proceed as if the non-heavy queries had been answered NO.

General Distributions D, Search Problems, One-Way Functions

So far we have described our results for the case in which D is a samplable distribution.

We show that a reduction of Impagliazzo and Levin [IL90] implies that for every distributional NP problem (L, D) and bound $\delta = 1/n^{O(1)}$ there is a non-adaptive probabilistic polynomial time oracle algorithm R, an NP language L', and a bound $\delta' = 1/n^{O(1)}$ such that for every oracle A that has agrees with L' on a $1-\delta'$ fraction of inputs, $R^{L'}()$ solves L on a subset of inputs of density $1-\delta$ under the distribution D.

This means that if there were a non-adaptive worst-case to average-case reduction with parameter $1/\text{poly}(n)$ from a problem L to a distributional problem (L', D), there would also be such a reduction from L to (L'', U), where U is the uniform distribution and L'' is in NP. By the previously described

[5]This is the way locally decodable codes are shown equivalent to smooth codes in [KT00].

[6]Formally, an honest prover succeeds with high probability if the probability of y is less than $t(1-\epsilon)/2^m$, and a cheating prover fails with high probability if the probability of y is more than $t/2^m$.

results, this would imply the collapse of the polynomial hierarchy.

A reduction by Ben-David and others [BCGL89] implies that for every distributional NP problem (L, U) there is a problem L' in NP such that an algorithm that solves the decision version of (L', U) on a $1 - \delta$ fraction of inputs can be modified (via a non-adaptive reduction) into an algorithm that solves the search version of (L, U) on a $1 - \delta \cdot \mathrm{poly}(n)$ fraction of input. This implies that even if modify the definition of worst-case to average-case reduction so that the oracle A is supposed to solve the *search* version of the problem, our results still apply. In particular, for every polynomial time computable function f, the problem of inverting f well on average is precisely the problem of solving well on average a distributional NP search problem. Therefore our results also rule out the possibility of basing one-way functions on NP-hardness using non-adaptive reductions.

2 Definitions and notation

We use functional and set notation for boolean functions interchangeably; say if $L : \{0,1\}^n \to \{0,1\}$, then "$x \in L$" is the same as "$L(x) = 1$".

By "k parallel instantiations" of an r round protocol P, we mean a protocol P' which creates k statistically independent instantiations $P_1, \ldots, P_k$ of P, and in its ith round runs the ith round of each P_j. At the end, P combines the outputs of the P_j according to a specified rule.

A *nonadaptive worst-case to average-case randomized reduction* from L to L' with *average hardness* δ (in short, a δ *worst-to-average reduction*) is a family of polynomial size circuits $R = \{R_n\}$ such that

1. On input $x \in \{0,1\}^n$, randomness r, $R_n(x; r)$ outputs strings $y_1, \ldots, y_k$ and a circuit A, called the *decoder*.
2. For any L^* that is δ-close to L', $\Pr_r[A(y_1, \ldots, y_k; L^*(y_1), \ldots, L^*(y_k)) = L(x)] > 2/3$.

Remarks.

1. The constant $2/3$ can be made $1 - 2^{-\Omega(k)}$ by parallel instantiation and taking majority at the end. We will assume this better bound from here on.
2. Without loss of generality, one may assume the number of strings $k = \mathrm{poly}(n)$ depends only on $n = |x|$, but not on the specific input x.
3. For notational convenience, we assume that all queries $y_1, \ldots, y_k \in \{0,1\}^m$ have the same length $m = \mathrm{poly}(n)$ that depends only on n but not on x. This assumption cannot be made without loss of generality, however all the proofs that we give can be generalized to the case of queries with different length, mostly by just replacing every mention of $\{0,1\}^m$ with the set of strings of length at most m, and every use of 2^{-m} with $2^{-m-1} + 1$.
4. One may also ask what happens to "Las Vegas" reductions that are only required to run in *expected* polynomial time. By standard tricks, it is not difficult to see that the existence of

δ worst-to-average Las Vegas reductions implies the existence of δ worst-to-average ordinary reductions.

We use ω to denote a function that grows faster than any constant, and we will abuse notation by writing expressions like $\omega + \omega = \omega$, $\omega^2 = \omega$, etc. "With high probability," or whp, means with probability $1 - o(1)$.

3 Preliminaries

Here we outline two protocols that will be used in the analysis.

3.1 The Goldwasser-Sipser Lower Bound Protocol

Given an NP set $S \subseteq \{0,1\}^n$ a bound s, we are interested in an AM protocol for the statement $|S| \geq s$. Consider the following protocol, due to Goldwasser and Sipser [GS86]:

1. Verifier: Choose a pairwise independent hash function $h : \{0,1\}^m \to \Gamma$, where $\Gamma = s/k$, and send h to the prover.
2. Prover: Send a list $r_1, \ldots, r_l \in \{0,1\}^m$.
3. Verifier: If $r_i \notin S$ for any i, reject. If $l < (1 - \epsilon/3)k$, reject. If $h(r_i) \neq 0$ for any i, reject. Otherwise, accept.

Lemma 1. *If $|S| \geq s$, there exists a prover that makes the verifier accept with probability $1 - 9/\epsilon^2 k$. If $|S| \leq (1 - \epsilon)s$, no prover makes the verifier accept with probability more than $9/\epsilon^2 k$.*

3.2 The Upper Bound Protocol

Suppose that the verifier of an AM protocol has access to a "secret" r, chosen uniformly at random from an NP set $S \subseteq \{0,1\}^m$. Can the verifier take advantage of her secret to verify a statement of the form $|S| \leq s$? Consider the following protocol, due to Aiello and Håstad [AH91]:

1. Verifier: Choose a 3-wise independent hash function $h : \{0,1\}^m \to \Gamma$, where $|\Gamma| = (s - 1)/k$ and send the pair $(h, h(r))$ to the prover.
2. Prover: Send a list $r_1, \ldots, r_l \in \{0,1\}^m$.
3. Verifier: If $r_i \notin S$ for any i, reject. If $l > (1 + \epsilon/3)k$ or $r \notin \{r_1, \ldots, r_l\}$, reject. Otherwise, accept.

Lemma 2. *If $|S| \leq s$, there exists a prover that makes the verifier accept with probability $1 - 9/\epsilon^2 k$. If $|S| \geq (1 + \epsilon)s$, no prover makes the verifier accept with probability $1 + 9/\epsilon^2 k - \epsilon/6$.*

Proof. Fix r and let $S' = S - \{r\}$, $\hat{k} = |S'|/|\Gamma|$, $R = h^{-1}(h(r))$, $R' = R - \{r\}$. For every $r' \in S'$, let $I_{r'}$ be an indicator for the event $r' \in R'$, so that $|R'| = \sum_{r' \in S'} I_{r'}$. By the 3-wise independence of h, the $I_{r'}$ are pairwise independent, so that $\mathrm{E}[|R'|] = \hat{k}$, $\mathrm{Var}[|R'|] = \hat{k}(1 - 1/|\Gamma|) < \hat{k}$ and by Chebyshev's inequality, for every $\xi > 0$:

$$\Pr[|R'| \notin (1 \pm \xi)\hat{k}] < 1/\xi^2 \hat{k}.$$

Suppose $|S| \leq s$. Without loss of generality we may assume $|S| = s$, since for larger values of s the acceptance probability may only increase. In this case $\hat{k} = k$, so that

$$\Pr[|R| > (1 + \epsilon/3)k] = \Pr[|R'| \geq (1 + \epsilon/3)k] < 9/\epsilon^2 k.$$

Given that $|R| \leq (1 + \epsilon/3)k$, the prover can list all elements of R, so that $R = \{r_1, \dots, r_l\}$ and $l \leq (1 + \epsilon/3)k$. In particular, this ensures that $r \in R$ and the verifier accepts.

Now suppose $|S| \geq (1 + \epsilon)s$, so that $\hat{k} > (1 + \epsilon)k$. Then

$$\Pr[|R'| < (1 + \epsilon/2)k] < \Pr\left[|R'| < \frac{1 + \epsilon/2}{1 + \epsilon}\hat{k}\right] < \Pr[|R'| < (1 - \epsilon/3)\hat{k}] < 9/\epsilon^2 \hat{k} < 9/\epsilon^2 k.$$

Given that $|R| > (1 + \epsilon/2)k$, what is the best strategy for a prover to make the verifier accept? Conditioned on $(h, h(r))$, r is uniformly distributed in R, so the best the prover can do is set $l = (1 + \epsilon/3)k$ and pick $\{r_1, \dots, r_l\}$ to be an *arbitrary* subset of R. In this case,

$$\Pr[r \in \{r_1, \dots, r_l\}] = l/|R| < \frac{(1 + \epsilon/3)k}{(1 + \epsilon/2)k} < 1 - \epsilon/6. \quad \square$$

At a first glance, this protocol may not seem very useful, as the completeness-soundness gap is very narrow. However, suppose we fix ϵ and want to apply t iterations of the protocol. Then choosing $k = \omega(t/\epsilon^2)$ will ensure that, with high probability, a good prover will never make the verifier reject. On the other hand, a crooked prover may cheat by more than an ϵ fraction only on about $O(1/\epsilon)$ of the t iterations. For t large enough, this becomes a negligible fraction of the total number of iterations. In our application of the protocol, we will be able to tolerate such a small fraction of errors.

4 Proof Outline

By a theorem of Boppana et al. [BHZ87], if coNP $\subseteq$ AM/poly, then $\Sigma_3 = \Pi_3$. Therefore, assuming $\Sigma_3 \neq \Pi_3$, the lack of a δ worst-to-average reductions from NP-hard L to some $L' \in$ NP will follow from the existence of an AM protocol for $\overline{L}$:

Theorem 1. *Let L be an* NP*-complete language under polynomial-time reductions, $L' \in$* NP*, $\delta = n^{-O(1)}$. If there is a δ worst-to-average reduction from L to L', then there is a* AM/poly *protocol for $\overline{L}$.*

Let R denote the reduction from the Theorem. Fix the input x, and let $\mathcal{R}_i : \{0,1\}^m \to [0,1]$ denote the distribution on the ith query produced by R on input x. We will use $|r|$ to denote the number of random bits used by the reduction. We use $R(x;\cdot)$ to denote the randomized computation which, on input x, outputs $y_1, \ldots, y_k$ and A. We call $R(x;\cdot)$ the *instantiation* of R on x.

Without loss of generality, we may assume that the distributions $\mathcal{R}_i$ are all equal. This is because the reduction R can apply a uniform random permutation to the queries $y_1, \ldots, y_k$, and have the decoder "disentangle" the permutation before it runs. Then the marginal distribution of every query becomes $(\mathcal{R}_1 + \ldots + \mathcal{R}_k)/k \triangleq \mathcal{R}$.

The coAM protocol for L will consist of three phases. In the first phase, we will look for a "threshold" $t^* = O(\delta^{-1})$ such that $\Pr_{y \sim \mathcal{R}}[\mathcal{R}(y) < t^* 2^{-m}]$ can be estimated within an inverse polynomial additive factor. In the second phase, we will use a "hiding protocol" to figure out a good estimate for the fraction of the queries y in L such that $\mathcal{R}(y) < t2^{-m}$. In the last phase, we will apply a variant of the Feigenbaum-Fortnow protocol for these queries.

Let $G : \{0,1\}^{|r|} \to \{0,1\}^m$ denote the circuit that, on input r, computes $R(x;r)$ and outputs the query y_1. When r is chosen uniformly at random, this circuit generates a query y sampled from $\mathcal{R}$.

5 The First Phase

Let $\Lambda(t) = \{y : \mathcal{R}(y) < t2^{-m}\}$ and $p(t) = \mathcal{R}(\Lambda(t))$. We think of $\Lambda(t)$ as a "ball of radius t." In this phase of the protocol, we look for a value of t such that a good lower bound on $p(t)$ can be obtained. We will estimate $p(t)$ by random sampling: Generate a sample $y \sim \mathcal{R}$ and test if $y \in \Lambda(t)$. Since there is no easy way to establish if $y \in \Lambda(t)$, we will take advantage of the prover for this purpose. The upper and lower bound protocols will ensure that the prover cannot cheat by much without getting caught.

First an easy technical lemma. Let $\epsilon = 1/\omega k$.

Lemma 3. *Fix an arbitrary sequence $0 < t_0 < t_1 < \ldots < t_l < 2^m$, where $l = \omega/\epsilon$. For i^* chosen uniformly at random from $\{1, \ldots, l\}$, with high probability, $p(t_{i^*}) \leq p(t_{i^*-1}) + \epsilon$.*

Proof. Let $I' = \{i \in I : p(t_i) > p(t_{i-1}) + \epsilon\}$. If $|I'| > 1/\epsilon$, then

$$p(t_l) > p(t_0) + 1/\epsilon \cdot \epsilon \geq 1.$$

Contradiction. It follows that $|I'| \leq 1/\epsilon$, so that with high probability, $i^* \notin I'$. □

We will apply the lemma to the sequence $t_i \triangleq \delta^{-1}(1 + \epsilon/\omega)^i$, so that $t_i = O(\delta^{-1})$ for $1 \leq i \leq \omega/\epsilon$.

We now present the first phase protocol:

1 Verifier: Let $l = \omega/\epsilon^3$. Independently choose $r_1, \ldots, r_l$ uniformly from $\{0,1\}^{|r|}$. Send $y_j \triangleq G(r_j)$ for $1 \leq j \leq l$ to the prover.

2 Prover: For each $1 \leq j \leq s$, send a claim ρ_j for the value $2^{|r|}\mathcal{R}(y_j)$.

3 Verifier: For each $1 \leq j \leq s$, initiate (in parallel) the upper bound protocol for the claim $|G^{-1}(y_j)| \leq \rho_j$ with parameter $k = \omega^2/\epsilon^5$. If any of the instances rejects, reject.

4 Verifier: For each $1 \leq j \leq s$, initiate (in parallel) the lower bound protocol for the claim $|G^{-1}(y_j)| \geq \rho_j$. If any of the instances reject, reject. Otherwise, choose i^* uniformly at random from $\{1, \ldots, \omega/\epsilon\}$, let $t^* = t_{i^*} = \delta^{-1}(1 + \epsilon/\omega)^{i^*}$ and set

$$p^* = \frac{|\{j : \rho_j 2^{-|r|} < t^* 2^{-m}\}|}{l}.$$

Lemma 4. *For every $x \in \{0,1\}^n$ there exists a "good" prover for which, with high probability, at the end of the Step 4, the verifier has not rejected.*

Lemma 5. *For every $x \in \{0,1\}^n$, and for any prover, with high probability, at the end of Step 3 either the verifier rejects or $p^* > p(t^*) - \epsilon$.*

Lemma 6. *For every $x \in \{0,1\}^n$, and for any prover, with high probability, at the end of Step 4 either the verifier rejects or $p^* < p(t^*) + \epsilon$.*

Proof. The analysis here is a bit simpler, since the lower bound protocol cannot cheat by "a lot": By Lemma 1, with probability $1 - 1/\omega l$, for any j,

$$\rho_j 2^{-|r|} > (1 - \epsilon/\omega)\mathcal{R}(y_j) > (1 - \epsilon/\omega)(1 + \epsilon/\omega)t^* 2^{-m} > t^* 2^{-m},$$

provided the verifier does not reject. With high probability, this is true for all j. The number of queries on which the prover cheats "a little" is bounded by 3ϵ with high probability, as in the previous proof. By Lemma 15, $|J_{i^*}| < p(t^*) + \epsilon$ whp. Putting this together:

$$p^* = \frac{|\{j : \rho_j 2^{-|r|} < t^* 2^{-m}\}|}{l} \leq \frac{|J_{i^*}|}{l} + \frac{|J_{i^*+1}| - |J_{i^*}|}{l} < (p(t^*) + \epsilon) + 3\epsilon. \quad \square$$

6 The Second Phase

In the second phase of the AM protocol, we try to obtain an estimate for $\mathcal{R}(L'')$, where L'' is the language $L' \cap \Lambda(t^*)$, i.e.,

$$L''(y) = \begin{cases} L'(y) & \text{if } \mathcal{R}(y) \leq t^* 2^{-m} \\ 0 & \text{otherwise.} \end{cases}$$

The language L'' is δ-close to L': The number of $y \in \{0,1\}^m$ such that $\mathcal{R}(y) > t^* 2^{-m}$ can be at most $t^{*-1}\delta 2^m \leq \delta 2^m$, since $\mathcal{R}$ is a probability distribution. Therefore $|L'' \oplus L'| \leq |\{0,1\}^m - \Lambda(t^*)| \leq \delta 2^m$.

Let $\mathcal{U}$ denote the uniform distribution on m bit strings. We assume that both the verifier and the prover know the probability $p_{adv} = \Pr_{y \sim \mathcal{U}}[y \in L']$. Let $\alpha = \delta/\omega$.

5 Verifier: Let $l = \omega/\alpha\epsilon^3$. Generate strings $y_1, \ldots, y_l \in \{0,1\}^m$ as follows: For every $1 \leq j \leq l$,

5.1 Toss a coin t_j, which is 1 with probability α, 0 with probability $1 - \alpha$.

5.2 If $t_j = 1$, choose $y_j \sim \mathcal{R}$. If $t_j = 0$, choose $y_j \sim \mathcal{U}$.

Let $T = \{j : t_j = 1\}$. Send the sequence $y_1, \ldots, y_l$ to the prover.

6 Prover: For each $1 \leq j \leq l$, send a claim $a_j \in \{0,1\}$ for the statement $y_j \in L'$. If $a_j = 1$, send an NP certificate for your claim. For each j, send a claim $b_j \in \{0,1\}$ for the statement $\mathcal{R}(y_j) \geq t^*$. Let $A^* = \{j : a_j = 1\}$, $B^* = \{j : b_j = 1\}$.

7 Verifier: For each $1 \leq j \leq l$, if $j \in B \cap T$, initiate the lower bound protocol for the claim $|G^{-1}(y_j)| \geq 2^{|r|}\mathcal{R}(y_j)$. Perform the following tests:

7.1 If $|A^* \cap \overline{T}|/|\overline{T}| < p_{adv} - \alpha\epsilon$, reject.

7.2 If $|\overline{B}^* \cap T|/|T| \notin p^* \pm \epsilon$, reject.

7.3 If any of the lower bound protocol instantiations fail, reject.

If all tests pass, define

$$q^* \triangleq \frac{|A^* \cap \overline{B}^* \cap T|}{|T|}.$$

Let $A = \{j : y_j \in L'\}$ and $\overline{B}_i = \{j : y_j \in \Lambda(t_i)\}$. By Lemma 15, with high probability $|\overline{T}| > (1 - \alpha - \epsilon)l$ and $|T| > (\alpha - \epsilon)l \geq \alpha l/2$.

Lemma 7. *For every $x \in \{0,1\}^n$ there exists a "good" prover for which with high probability, the verifier has not rejected by the end of Step 7 and $q^* < \mathcal{R}(L'') + \epsilon$.*

Proof. The good prover sends correct claims for all the queries; it gives answers $a_j = L(y_j)$ and $b_j = 1$ iff $\mathcal{R}(y_j) \geq t^*$. We show this prover is likely to pass all tests, using Lemma 15 on several occasions.

Test 7.1: Follows directly from Lemma 15.

Test 7.2: With high probability, $|\overline{B}^* \cap T| \in (p(t^*) \pm \epsilon)|T|$. By Lemmas 5 and 6, $p(t^*) \in p^* \pm \epsilon$ with high probability, so that $|\overline{B}^* \cap T|/|T| \in p^* \pm \epsilon$.

Test 7.3: By Lemma 1, this test passes with high probability.

It remains to show that $q^* < \mathcal{R}(L'') + \epsilon$. First, $A^* \cap \overline{B}^* = A \cap \overline{B}_{i^*}$, so that q^* is an unbiased estimator for the fraction of queries in T that fall into $L \cap \Lambda(t^*) = L'$, when the queries are drawn from $\mathcal{R}$. Since the number of samples in T is at least $\alpha l/2 > \omega/\epsilon^3$, $q^* < \mathcal{R}(L') + \epsilon$. ☐

Lemma 8. *For every $x \in \{0,1\}^n$, and for any prover, with high probability, the verifier either rejects by the end of Step 7 or $q^* > \mathcal{R}(L'') - \epsilon$.*

First, note that the verifier cannot make any false "yes" claims; if $a_j = 1$, it must be that $y_j \in L$, otherwise the prover will detect a faulty NP certificate for y_j. So the verifier can only cheat by making false "no" claims. Let $C = A - A^*$ denote the set of indices corresponding to these claims.

The main idea of the proof is to show that if $|C \cap \overline{B}^* \cap T|$ is a significant fraction of $|T|$, the verifier is likely to reject. Suppose the opposite is true, i.e., the prover cheats on many queries in T. We will show that the prover cannot distinguish, with significant confidence, whether a query in $\overline{B}^*$ came from T or from $\overline{T}$; so if he cheats on many queries in $\overline{B}^* \cap T$, he will also end up cheating on a lot of queries in $\overline{B}^* \cap \overline{T} \subseteq \overline{T}$. But in this case the prover will get caught in Step 7.1.

Lemma 9. *For any choice of C made by the prover in Step 5, if $|C \cap \overline{B}_{i^*}| > 6\alpha\epsilon l$, then with high probability, $|C \cap \overline{T}| > 2\alpha\epsilon l$.*

Proof. First we show that whenever $j \in \overline{B}_{i^*}$, the prover cannot tell if $t_j = 1$ based on its evidence with confidence over 1/2:

$$\begin{aligned}\Pr[j \in T | y_1, \ldots, y_l] &= \Pr[j \in T | y_j] \\ &= \frac{\Pr[y_j | j \in T] \Pr[j \in T]}{\Pr[y_j]} \\ &\leq \frac{\Pr[y_j | j \in T] \Pr[j \in T]}{\Pr[y_j | j \notin T] \Pr[j \notin T]} \\ &< \frac{\omega\delta^{-1} 2^{-m} \cdot \alpha}{2^{-m} \cdot (1-\alpha)} = \frac{1}{2}.\end{aligned}$$

Even when conditioned on seeing $y_1, \ldots, y_l$, the events "$j \in T$", where $j \in C \cap \overline{B}_{i^*}$, are independent. By a crude estimate, with high probability, $|C \cap \overline{B}_{i^*} \cap T| < 4\alpha\epsilon l$, so that $|C \cap \overline{T}| \geq |C \cap \overline{B}_{i^*} \cap \overline{T}| > 2\alpha\epsilon l$. ∎

Lemma 10. *For every $x \in L$, and any prover, if the verifier survives Step 7, then $|(\overline{B}^* \oplus \overline{B}_{i^*}) \cap T| < \epsilon|T|$ with high probability.*

Proof. Suppose the verifier survives Step 7. By Lemma 15, with high probability, $|\overline{B}_{i^*} \cap T| \in (p(t^*) \pm \epsilon)|T|$ and $|\overline{B}_{i^*-1} \cap T| \in (p(t_{i^*-1}) \pm \epsilon)|T|$. By Lemma 3, with high probability, $p(t^*) < p(t_{i^*-1}) + \epsilon$. Putting this together, $|\overline{B}_{i^*} \cap T| < |\overline{B}_{i^*-1} \cap T| + 3\epsilon|T|$.

First we show that $|(\overline{B}_{i^*} - \overline{B}^*) \cap T| < 3\epsilon|T|$. By Lemma 1, with high probability, $\overline{B}_{i^*-1} \cap T \subseteq \overline{B}^* \cap T$, for otherwise the verifier wouldn't survive Step 7.3. It follows that

$$|(\overline{B}_{i^*} - \overline{B}^*) \cap T)| \leq |\overline{B}_{i^*-1} - \overline{B}^* \cap T)| + |\overline{B}_{i^*} - \overline{B}_{i^*-1} \cap T| = |\overline{B}_{i^*} \cap T| - |\overline{B}_{i^*-1} \cap T| < 3\epsilon|T|.$$

Now we show that $|(\overline{B}^* - \overline{B}_{i^*}) \cap T| < \epsilon|T|$. Since the verifier survives Step 7.2, by Lemma 5 with high probability:

$$|\overline{B}^* \cap T| \leq (p^* + \epsilon)|T| \leq (p(t^*) + \epsilon)|T|.$$

On the other hand,

$$|\overline{B}_{i^*} \cap \overline{B}^* \cap T| = |\overline{B}_{i^*} \cap T| - |(\overline{B}_{i^*} - \overline{B}^*) \cap T| > (p(t^*) - \epsilon)|T| - 3\epsilon|T| = (p(t^*) - \epsilon)|T|,$$

so that

$$|(\overline{B}^* - \overline{B}_{i^*}) \cap T| = |\overline{B}^* \cap T| - |\overline{B}_{i^*} \cap \overline{B}^* \cap T| < (p(t^*) + \epsilon)|T| - (p(t^*) + \epsilon)|T| < \epsilon|T|. \quad ∎$$

Proof of Lemma 8. If $|C \cap \overline{B}_{i^*}| > 6\alpha\epsilon l$, then by Lemma 9 $|C \cap \overline{T}| > 2\alpha\epsilon|\overline{T}|$ with high probability. By Lemma 15, $|A \cap \overline{T}| < (p_{adv} + \alpha\epsilon)|\overline{T}|$ with high probability. Next, $|A^* \cap \overline{T}| = |A \cap \overline{T}| - |C \cap \overline{T}| < (p_{adv} - \alpha\epsilon)|\overline{T}|$, and Step 7.1 rejects.

Suppose $|C \cap \overline{B}_{i^*}| \leq 6\alpha\epsilon l$. Then

$$\begin{aligned}|A^* \cap \overline{B}^* \cap T| &\geq |A^* \cap \overline{B}_{i^*} \cap T| - |(\overline{B}^* \oplus \overline{B}_{i^*}) \cap T| \\ &\geq (|A \cap \overline{B}_{i^*} \cap T| - |C \cap \overline{B}_{i^*}|) - |(\overline{B}^* \oplus \overline{B}_{i^*}) \cap T| \\ &\geq (\mathcal{R}(L \cap \Lambda(t^*)) - \epsilon)|T| - 6\alpha\epsilon l - \epsilon|T| \\ &\geq (\mathcal{R}(L') - \epsilon)|T|. \end{aligned}$$

□

7 The Third Phase

The third phase is a variation of the Feigenbaum-Fortnow protocol for reductions with uniform marginal distributions.

8 Verifier: Let $l = \omega \log k/\epsilon^3$. Run l independent instances of the reduction $R_n(x; \cdot)$. Say the ith instance produces the circuit A^i and queries $y_{i1}, \ldots, y_{ik} \in \{0,1\}^m$. Send the queries y_{ij}, $1 \leq i \leq l, 1 \leq j \leq k$ to the prover.

9 Prover: For each pair i, j, send a claim a_{ij} for the statement $y_{ij} \in L'$ (accompanied by an NP certificate, if $a_{ij} = 1$) and a claim b_{ij} for the statement $\mathcal{R}(y_{ij}) \geq t^*$. Let $A_j^* = \{i : a_{ij} = 1\}, B_j^* = \{i : b_{ij} = 1\}$.

10 Verifier: For each pair i, j, if $b_{ij} = 1$, initiate the lower bound protocol for the claim $|\{r : G(r) = y_{ij}\}| \geq 2^{|r|}t^*$. Let $c_{ij} = a_{ij}(1 - b_{ij})$. Perform the following tests:

10.1 If for any j, $|\overline{B}_j^*|/l \notin p^* \pm \epsilon$,
reject.

10.2 If for any j, $|A_j^* \cap \overline{B}_j^*|/l < q^* - \epsilon$, reject.

10.3 If for any i, $A^i(y_{i1}, \ldots, y_{ik}; c_{i1}, \ldots, c_{ik})$ accepts, reject.

If all tests pass, accept.

Let $A_j = \{i : y_{ij} \in L\}$, $\overline{B}_j = \{i : \mathcal{R}(y_{ij}) < t^*\}$.

Lemma 11. *For every $x \notin L$, there exists a "good" prover that accepts with high probability by the end of Step 10.*

Proof. The good prover claims $a_{ij} = L'(y_{ij})$ and $b_{ij} = 1$ iff $\mathcal{R}(y_{ij}) \geq t^*$, for all pairs i, j. By Lemmas 5, 6 and 7, we may assume $p^* \in p(t^*) \pm \epsilon$ and $q^* < \mathcal{R}(L') + \epsilon$ with high probability. Consider the Step 10 tests:

Test 10.1: $|\overline{B}_j^*|/l = |\overline{B}_j|/l \in p(t^*) \pm \epsilon \in p^* \pm \epsilon$, with probability $1/\omega k$ (Lemma 15). Therefore Test 10.1 passes for all j with high probability.

Test 10.2: With probability $1/\omega k$, $|A_j^* \cap \overline{B}_j^*|/l = |A_j \cap \overline{B}_j|/l < \mathcal{R}(L') - \epsilon < q^* - \epsilon.$, so Test 11.2 passes for all j with high probability.

Test 10.3: Note that $c_{ij} = 1$ iff $y_{ij} \in L'$, so that $A^i(y_{i1}, \ldots, y_{ik}; c_{i1}, \ldots, c_{ik}) = L(x) = 0$ with probability $1/\omega l$. So Test 11.3 passes for all i with high probability. □

Lemma 12. *For every $x \in L$, and for any prover, with high probability, the verifier rejects by the end of Step 10.*

Proof. Assume the verifier passes all instances of Test 11.1 and Test 11.2. We show that, with high probability, Test 11.3 must reject for some i. There are two types of queries for which the prover can fool the verifier about membership in L': First, there are the queries that fall into $\overline{B}_j^* \oplus \overline{B}_j$, for which b_{ij} may be a lie. Then there are the queries in $|(A_j - A_j^*) \cap \overline{B}_j^*|$, for which $b_{ij} = 0$ but a_{ij} may be a lie. Let C_j be the set of all possible lies:

$$C_j = (\overline{B}_j^* \oplus \overline{B}_j) \cup ((A_j - A_j^*) \cap \overline{B}_j^*).$$

As in the Feigebaum-Fortnow proof, the main idea is to show that for any fixed j, $|C_j|/l < 1/k$, so that there exists at least one $i \notin C_1 \cup \ldots \cup C_k$. For this i, it will follow that $c_{ij} = L'(y_{ij})$ for all j, so that with high probability, $A(y_{i1}, \ldots, y_{ik}; c_{i1}, \ldots, c_{ik}) = L(x) = 1$ and Test 11.3 rejects.

We bound $|\overline{B}_j^* \oplus \overline{B}_j|$ in the same fashion as in Lemma 10; by that argument, we have $|\overline{B}_j^* \oplus \overline{B}_j| < \epsilon l < l/2k$ with probability $1 - 1/\omega k$. For the other term,

$$\begin{aligned}
|(A_j - A_j^*) \cap \overline{B}_j^*| &= |A_j \cap \overline{B}_j^*| - |A_j^* \cap \overline{B}_j^*| \\
&< (|A_j \cap \overline{B}_j| + |\overline{B}_j \oplus \overline{B}_j^*|) - |A_j^* \cap \overline{B}_j^*| \\
&< (\mathcal{R}(L') + \epsilon)l + \epsilon l) - (q^* - \epsilon)l \\
&= (\mathcal{R}(L') - q^*)l + \epsilon l \\
&< \epsilon l \text{ whp, by Lemma 8.}
\end{aligned}$$

□

8 Average-case complexity for arbitrary samplable distributions

Let V be an NP-relation. We denote by L_V the NP-language corresponding to V, i.e., $L_V(x) = 1$ iff there exists a w such that $V(x, w) = 1$. A family of random functions $F_n : \{0,1\}^n \to \{0,1\}^m$ is a δ-*approximate witness oracle* for V with respect to distribution $\mathcal{D}$ if for all n,[7]

$$\Pr_{x \sim \mathcal{D}, F}[V(x, F_{|x|}(x)) = L_V(x)] > 1 - \delta.$$

We will omit the subscript of F when it is implicitly determined by the input length. Note that the definition implies the existence of a set S of measure $\mathcal{D}(S) = 1 - 3\delta$ and for all $x \in S$,

$$\Pr_F[V(x, F_{|x|}(x)) = L_V(x)] > 2/3.$$

[7] Technically, a witness oracle is a distribution over function families $\{F_n\}$, but to simplify notation we will identify samples from this distribution with the distribution itself.

Intuitively, S is the set of inputs where the oracle has a good chance of producing a witness for the input. As usual, the constant $2/3$ is arbitrary, since if one has access to F, it can be queried k times independently in parallel to obtain a good witness with probability $1 - 1/3^k$.

Just as languages in NP represent decision problems, witness oracles represent search problems. For example, inverting a one-way function $f : \{0,1\}^n \to \{0,1\}^n$ on a $1 - \delta$ fraction of inputs amounts to finding an algorithm $A : \{0,1\}^n \to \{0,1\}^n$ that is δ-approximate for the relation $V(y, x) \iff y = f(x)$ with respect to the distribution $f(\mathcal{U})$.

Using witness oracles, we can formalize the notion of nonadaptive reductions between search problems, as well as reductions from search to decision problems, and vice-versa. Let V, V' be NP relations and $\mathcal{D}, \mathcal{D}'$ be arbitrary polynomial-time samplable distributions. A *δ-to-δ' average-to-average reduction* for search problems from $(V, \mathcal{D})$ to $(V', \mathcal{D}')$ is a family of polynomial-size circuits $R = \{R_n\}$ such that:

1. On input $x \in \{0,1\}^n$, randomness r, $R_n(x; r)$ outputs strings $y_1, \ldots, y_k$ and a "decoder" circuit A.
2. For any witness oracle F^* that is δ'-approximate for V' with respect to $\mathcal{D}'$,

$$\Pr_{x \sim \mathcal{D}, F^*}[V(x, A(y_1, \ldots, y_k; F^*(y_1), \ldots, F^*(y_k))) = L_V(x)] > 1 - \delta.$$

The other two types of reductions are defined in similar fashion. A δ' worst-to-average reduction is a 0-to-δ' average-to-average reduction. Using this terminology, we state and prove the conditional impossibility of nonadaptive worst-to-average reductions in its most general form:

Theorem 2. *Let L be a language that is* NP*-hard under polynomial-time reductions, V' be an* NP*-relation, $\mathcal{D}'$ be an arbitrary polynomial-time samplable distribution, and $\delta = n^{-O(1)}$. If there is a δ worst-to-average reduction from L to V', then there is an* AM/poly *protocol for $\overline{L}$.*

The theorem is an immediate consequence of Theorem 1 and the following two lemmas:

Lemma 13. *For every* NP*-relation $V \subseteq \{0,1\}^n \times \{0,1\}^m$ (where $m = n^{O(1)}$) there exists an* NP*-language L' and a constant c such that there is a $O(\delta m^2)$-to-δ average-to-average reduction from $(V, \mathcal{U})$ to $(L', \mathcal{U})$.*

Lemma 14. *For every* NP*-relation V and polynomial-time samplable distribution $\mathcal{D} = D(\mathcal{U})$, where $D : \{0,1\}^{|r|} \to \{0,1\}^n$, $|r| = n^{O(1)}$, there exists an* NP*-relation V' such that there is a $O(\delta|r|)$-to-δ average-to-average reduction from $(V, \mathcal{D})$ to $(V', \mathcal{U})$.*

Analogues of these lemmas are known in the context of the distributional hardness of NP-problems. A variant of Lemma 13 appears Ben-David et al. [BCGL89], while a variant of Lemma 14 was proved by Impagliazzo and Levin [IL90]. Our proofs are in essence a recasting of these arguments in the formalism of nonadaptive average-to-average reductions.

Proof of Lemma 13. As in [BCGL89], we first reduce the search problem V to a search problem with a unique witness, then encode the bits of the witness in the language L'. The first step is based on the hashing argument of Valiant and Vazirani [VV86]. The reduction, as described below, only succeeds with probability 1/16, but this can be amplified to 2/3 by applying the reduction say 100 times in parallel.

Let $a[i]$ denote the ith bit of a binary string a. Given $a, b \in \{0,1\}^n$, define $a \cdot b = \sum_{i=1}^n a[i]b[i] \mod 2$. We specify the language L as follows: Given $x \in \{0,1\}^n, a_1, \ldots, a_m \in \{0,1\}^m, k, j \in [m]$, let $(x, c_1, \ldots, c_m, k, j) \in L'$ if there exists a $w \in \{0,1\}^m$ such that $V(x, w) = 1$ and $w \cdot c_i = 0$ for $1 \leq i \leq k$ and $w[j] = 1$. It is immediate that $L \in \mathrm{NP}$.

The reduction works as follows: On input $x \in \{0,1\}^n$, independently choose $c_1, \ldots, c_m \sim \mathcal{U}(\{0,1\}^m)$ and generate the queries $q_{kj} = (x, c_1, \ldots, c_m, k, j)$ for all $1 \leq k \leq m$ and $1 \leq j \leq m$. Let $a_{kj} \in \{0,1\}$ denote the claimed answer to query q_{kj} and w_k be the concatenation $a_{k1} \ldots a_{km}$. The decoder looks for an index k such that w_k is a witness for $(x, a_1, \ldots, a_m, k, j)$ for all $1 \leq j \leq m$ and outputs w_k; if no such k is found the decoder returns the string 0^m.

Let L^* be an arbitrary decision oracle that is δ-close to L'. Say an input x is *good* in L^* if for all $1 \leq k \leq m$, $1 \leq j \leq m$,

$$\Pr_{c_1, \ldots, c_m, r}[L^*(x, c_1, \ldots, c_m, k, j; r) = L(x, c_1, \ldots, c_m, k, j)] > 15/16.$$

By a pigeonhole argument, $x \sim \mathcal{U}_n$ is good with probability at least $1 - O(\delta m^2)$. We show that the reduction succeeds on a good input with probability 1/16. By the Valiant-Vazirani argument, for $k = \log_2 |\{w : V(x; w)\}|$, with probability 1/8 there exists a unique w such that $w \cdot c_i = 0$ for $1 \leq i \leq k$. It follows that whenever x is good and $x \in L_V$, with probability at least 1/16, $L^*(x, c_1, \ldots, c_m, k, j; r) = w[j]$ for all $1 \leq j \leq m$, so the decoder encounters the witness w. $\square$

Remark. The argument can be strengthened to obtain a $O(\delta m)$-to-δ average-to-average reduction from (V, U) to (L', U) by applying a Hadamard code to the witness w in L' instead of revealing its bits.

Proof of Lemma 14. Let V' be an NP-relation for language L', whose inputs are

1. An integer $k \in [|r|]$, which will encode the entropy of a string from $\mathcal{D}$,
2. A pairwise independent hash function $h_1 : \{0,1\}^n \to \{0,1\}^{k+6}$; for example, $h_1(x) = Ax + b$, where $A \in \{0,1\}^{n \times (k+6)}, b \in \{0,1\}^{k+6}$,
3. A pairwise independent hash function $h_2 : \{0,1\}^{|r|} \to \{0,1\}^{|r|-k-3}$, for example $h_2(r) = Ar + b$, $A \in \{0,1\}^{|r| \times (|r|-k-3)}, b \in \{0,1\}^{|r|-k-3}$,
4. A string $z \in \{0,1\}^{k+6}$,
5. A padding string $p \in \{0,1\}^{n|r|+(|r|-n-1)k}$, so that the length of the input depends only on n (and $|r|$) but not on k.

A pair (w, r) is an NP-witness for input (k, h_1, h_2, z, p) in V' if $V(D(r); w) = 1$ and $h_1(D(r)) = z_1$ and $h_2(r) = 0$.

On input x, the reduction produces queries $(k, h_1, h_2, h_1(x), p)$, for all possible values of k by choosing h_1, h_2 and p uniformly at random from their range. The decoder looks at all answers (w_k, r_k), and returns w_k if $V(D(r_k); w_k) = 1$ for some k. If no such k is found, the decoder returns the string 0^m.

Suppose F^* is a δ-approximate oracle for V' with respect to the uniform distribution. Given $x \in \{0,1\}^n$, we call an instance (k, h_1, h_2, z, p) *good* for x if the following three conditions are satisfied:

1. $k = \lfloor -\log_2 \mathcal{D}(x) \rfloor$ and $z = h_1(x)$
2. There exists an r such that $h_1(D(r)) = z$ and $h_2(r) = 0$
3. If, for some r, $h_1(D(r)) = z$ and $h_2(r) = 0$, then $D(r) = x$.

Let $G(x)$ denote the set of all queries in L' that are good for x. It is immediate that the sets $G(x)$ are pairwise disjoint over all $x \in \{0,1\}^n$. On the one hand, we will show that, on input x, the reduction has a constant probability of producing a query that lands in $G(x)$. Moreover, conditioned on k, this query is uniformly distributed in $G(x)$. If $x \in L$ and F^* and V' agree on the query that falls within $G(x)$, then $F^*(x) = (w, r)$ with $D(r) = x$, so $V(x; w) = 1$. On the other hand, we will show that when $x \sim \mathcal{D}$, with probability at least $1 - \delta|r|$, F^* and V' do agree on a constant fraction of $G(x)$ for every k, so that the reduction has a constant probability of producing a query on which F^* and V' agree.

Claim 1. *Let $x \in \{0,1\}^n, k = \lfloor -\log_2 \mathcal{D}(x) \rfloor$. With probability $3/4$ over the choice of h_1, h_2 and p, the instance $(k, h_1, h_2, h_1(x), p)$ is in $G(x)$.*

Proof of Claim. We first show that, with probability 7/8, the instance satisfies the second condition for goodness, i.e., there exists an r such that $D(r) = x$ and $h_2(r) = 0$. For $r \in D^{-1}(x)$, let I_r be an indicator for the event $h_2(r) = 0$. By our choice of k, $|D^{-1}(x)| \geq 2^{|r|-k}$, so that

$$\mathrm{E}[\sum_{r \in D^{-1}(x)} X_r] \geq 2^{|r|-k}\mathrm{E}[I_r] = 8.$$

As the I_r are pairwise independent, the variance of this sum is at most the expectation, so by Chebyshev's inequality at least one $I_r = 1$ with probability 7/8.

Now we look at the probability of satisfying the third condition for goodness. Fix r such that $D(r) \neq x$. By pairwise independence, $\Pr[h_1(D(r)) = h_1(x)] = 2^{-k-6}$, and independently, $\Pr[h_2(r) = 0] = 2^{-|r|+k+3}$. It follows that

$$\begin{aligned}\Pr[\exists r : D(r) \neq x \wedge h_1(D(r)) = h_1(x) \wedge h_2(r) = 0] &\leq \sum_{r \notin D^{-1}(x)} \Pr[h_1(D(r)) = h_1(x)] \Pr[h_2(r) = 0] \\ &\leq \sum_{r \in \{0,1\}^{|r|}} 2^{-k-6} 2^{-|r|+k+3} = 1/8.\end{aligned}$$

It follows that both conditions for goodness are satisfied with probability at least 3/4. $\square$

Claim 2. *For every* $x \in \{0,1\}^n$*,* $\mathcal{U}(G(x)) = \Omega(\mathcal{D}(x)/|r|)$*.*

Proof of Claim. Consider a random string (k, h_1, h_2, z, p). With probability $1/|r|$, $k = \lfloor -\log_2 \mathcal{D}(x) \rfloor$. Conditioned on this, $z = h_1(x)$ with probability 2^{-k-3}. By the last Claim, with probability $3/4$ over h_1, h_2, p, $(k, h_1, h_2, h_1(x), p)$ is in $G(x)$. Putting this together,

$$\Pr[(k, h_1, h_2, z, p) \in G(x)] \geq \frac{1}{|r|} \cdot \frac{3}{4} \cdot 2^{-k-3} = \Omega(\mathcal{D}(x)/|r|). \qquad \square$$

Let S denote the set of all $x \in L_V$ for which

$$\Pr_{y \sim \mathcal{U}, F^*}[V'(y, F^*(y)) = 1 | y \in G(x)] > 8/9,$$

so that if the k-th query q_k lands into $G(x)$, the answer (w_k, r_k) has a $8/9$ probability of being a good witness for the query. It follows that, unconditionally, $V(q_k, F^*(q_k)) = 1$ with probability at least $8/9 \cdot 3/4 = 2/3$, and the decoder is successful on the queries that come from S.

On the other hand, by the disjointness of the sets $G(x)$,

$$\begin{aligned}
\delta &\geq \sum_{x \in L_V} \mathcal{U}(G(x)) \Pr_{y \sim \mathcal{U}}[V'(y, F^*(y)) = 0 | y \in G(x)] \\
&> \sum_{x \in \overline{S}} \mathcal{U}(G(x)) \cdot \frac{1}{9} \\
&= \sum_{x \in \overline{S}} \Omega(\mathcal{D}(x)/|r|) \text{ by Claim 2} \\
&= \Omega(\mathcal{D}(\overline{S})/|r|),
\end{aligned}$$

so that $\mathcal{D}(\overline{S}) = O(\delta |r|)$. $\square$

9 Acknowledgements

We thank Madhu Sudan for suggesting the relevance of [IL90], Oded Goldreich for stressing the relevance of our result to the question of basing cryptography on NP-hardness, and Amit Sahai for helpful discussions. The hiding protocol was suggested by Manikandan Narayanan.

References

[AD97] Miklós Ajtai and Cynthia Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In *Proceedings of the 29th ACM Symposium on Theory of Computing*, pages 284–293, 1997.

[AH91] W. Aiello and J. Håstad. Statistical zero-knowledge languages can be recognized in two rounds. *Journal of Computer and System Sciences*, 42:327–345, 1991.

[Ajt96] Miklós Ajtai. Generating hard instances of lattice problems. In *Proceedings of the 28th ACM Symposium on Theory of Computing*, pages 99–108, 1996.

[BCGL89] Shai Ben-David, Benny Chor, Oded Goldreich, and Michael Luby. On the theory of average-case complexity. In *Proceedings of the 21st ACM Symposium on Theory of Computing*, pages 204–216, 1989.

[BHZ87] Ravi Boppana, Johan Håstad, and Spyros Zachos. Does coNP have short interactive proofs? *Information Processing Letters*, 25:127–132, 1987.

[BK95] Manuel Blum and Sampath Kannan. Designing programs that check their work. *Journal of the ACM*, 41(1):269–291, 1995. Also in STOC'89.

[BLR93] Manuel Blum, Michael Luby, and Ronitt Rubinfeld. Self-testing/correcting with applications to numerical problems. *Journal of Computer and System Sciences*, 47(3):549–595, 1993.

[Blu88] Manuel Blum. Designing programs to check their work. Technical Report 88-09, ICSI, 1988.

[Bra79] Gilles Brassard. Relativized cryptography. In *Proceedings of the 20th IEEE Symposium on Foundations of Computer Science*, pages 383–391, 1979.

[DH76] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, I22(6):644–654, 1976.

[FF93] Joan Feigenbaum and Lance Fortnow. On the random-self-reducibility of complete sets. *SIAM Journal on Computing*, 22:994–1005, 1993.

[GG98] Oded Goldreich and Shafi Goldwasser. On the possibility of basing cryptography on the assumption that $P \neq NP$. Unpublished manuscript, 1998.

[GKST02] Oded Goldreich, Howard Karloff, Leonard Schulman, and Luca Trevisan. Lower bounds for linear locally decodable codes and private information retrieval. In *Proceedings of the 17th IEEE Conference on Computational Complexity*, pages 175–183, 2002.

[GS86] Shafi Goldwasser and Michael Sipser. Private coins versus public coins in interactive proof systems. In *Proceedings of the 18th ACM Symposium on Theory of Computing*, pages 59–68, 1986.

[IL90] Russell Impagliazzo and Leonid Levin. No better ways to generate hard NP instances than picking uniformly at random. In *Proceedings of the 31st IEEE Symposium on Foundations of Computer Science*, pages 812–821, 1990.

[Imp95] Russell Impagliazzo. A personal view of average-case complexity. In *Proceedings of the 10th IEEE Conference on Structure in Complexity Theory*, pages 134–147, 1995.

[KT00] Jonathan Katz and Luca Trevisan. On the efficiency of local decoding procedures for error correcting codes. In *Proceedings of the 32nd ACM Symposium on Theory of Computing*, pages 80–86, 2000.

[Lev86] Leonid Levin. Average case complete problems. *SIAM Journal on Computing*, 15(1):285–286, 1986.

[Lip89] Richard Lipton. New directions in testing. In *Proceedings of DIMACS Workshop on Distributed Computing and Cryptography*, 1989.

[O'D02] Ryan O'Donnell. Hardness amplification within NP. In *Proceedings of the 34th ACM Symposium on Theory of Computing*, pages 751–760, 2002.

[Reg03] Oded Regev. New lattice based cryptographic constructions. In *Proceedings of the 35th ACM Symposium on Theory of Computing*, pages 407–416, 2003.

[VV86] Leslie Valiant and Vijay Vazirani. NP is as easy as detecting unique solutions. *Theoretical Computer Science*, 47(1):85–93, 1986.

A Additive bounds for random sampling

Lemma 15. *Let $\epsilon < 1$, $T \subseteq \Omega$, $\mathcal{R}$ a distribution on Ω, $\mathcal{R}(T) = p$ and S an $N > 3\log(\eta/2)/\epsilon^3$ element random sample from Ω, drawn from $\mathcal{R}$. With probability $1-\eta$, $|S \cap T|/N \in p \pm \epsilon$.*

Proof. We use the following form of the Chernoff bound:

$$\Pr[|S \cap T| < (1-\xi)Np] < \exp(-\xi^2 Np/3), \text{for } \xi < 1$$

and

$$\Pr[|S \cap T| > (1+\xi)Np] < \begin{cases} 2^{-(1+\xi)Np}, & \text{for } \xi \geq 1, \\ \exp(-\xi^2 Np/3), & \text{for } \xi < 1. \end{cases}$$

If $p < \epsilon$, the lower bound holds trivially, and for the upper bound we set $\xi = \epsilon/p > 1$ to obtain

$$\Pr[|S \cap T| > (p+\epsilon)N] < 2^{-(p+\epsilon)N} < \eta.$$

If $p \geq \epsilon$, we set $\xi = \epsilon$ to obtain:

$$\Pr[|S \cap T| \notin (1 \pm \epsilon)Np] < 2\exp(-\epsilon^2 Np/3) \leq 2\exp(-\epsilon^3 N/3) < \eta. \quad \square$$

In most applications here we set $\eta = o(1)$, so that the estimate holds with high probability.