2011 26th Annual IEEE Conference on Computational Complexity

# Relativized Separations of Worst-Case and Average-Case Complexities for $NP$

Russell Impagliazzo *
*U. C. San Diego and IAS*
russell@cs.ucsd.edu

***Keywords***-P vs. NP, average-case complexity, oracles, relativization, heuristica

**Abstract**— Non-relativization of complexity issues can be interpreted as showing that these issues cannot be resolved by "black-box" techniques. We show that the assumption $DistNP \subseteq AvgP$ does not imply that $NP \subseteq BPP$ by relativizing techniques. More precisely, we give an oracle relative to which the assumption holds but the conclusion fails. Moreover, relative to our oracle, there are problems in $NP \cap Co - NP$ that require exponential circuit complexity. We also give an alternate version where $DistNP \subseteq AvgP$ is true, but a problem in the second level of the polynomial hierarchy is hard on the uniform distribution.

## 1. Introduction

Intractability is a two-edged sword. On the negative side, it is a barrier to solving problems efficiently. On the positive side, intractability has applications such as for cryptography and derandomization. Unfortunately, there are a number of gaps in making this situation "win-win". One of them is that worst-case performance is the gold standard for algorithm design, but most of the applications of intractability require average-case hardness on a known, simple distribution. This raises the question of whether worst-case hardness for $NP$ problems implies average-case hardness for such problems. We give the first relativization of this question, constructing an oracle relative to which all $NP$ problems are average-case tractable on all polynomial-time samplable input distributions, yet there are $NP$ problems which are intractable in the worst-case. (In fact, we get a much stronger worst-case intractability than merely $P \neq NP$. Relative to our oracle, there is a problem in $UP \cap co - UP$ that requires exponential sized circuit complexity.) Moreover, we show that there exist such relativized worlds where problems in the polynomial-time hierarchy are hard in the average-case.

It is widely believed that many problems in $NP$ are, in fact, hard in the average-case. Thus, the implication to which we give a relativized counter-example is likely to be true. However, our result is evidence that the implication is likely to be non-trivial to prove. Many worst-case to average-case reductions are known, both for complexity classes above $NP$ such as $PSPACE$ or $\#P$, and for specific problems within subclasses of $NP$ such as approximating the shortest vector in a lattice. All of these results for larger complexity classes relativize. (While the original proofs of some of these results were phrased in terms of specific complete problems for the classes, they can easily be rephrased using general closure properties that relativize.) The results for specific problems in $NP$ are ambiguous in interpretation. The problems are just as hard on average as they are in the worst-case, but there is no complexity-theoretic reason to believe they are in fact hard in the worst-case. So their random self-reducibility may be an indication that they are actually in $BPP$, not that they are hard on average.

These two kinds of results naturally lead to the question of whether similar random-self-reducibilities can be found for $NP$-complete problems. The random-self-reductions that have been found for problems such as shortest vectors in a lattice could also be seen to put the problem in $Co - AM$, so is unlikely to extend to $NP$-complete problems. More generally, there are several results about the limitations of randomized reductions ([FF93], [BT03], [AGGM06]), showing that the existence of certain types of worst-case to average-case reductions for a language $L \in NP$ imply that $L$ is in a sub-class of $NP$ believed to be proper. This shows that most arguments that use the specific nature of a language to give a worst-case / average-case connection (like that for lattice problems) cannot apply to $NP$-complete problems (unless, e.g., the polynomial-time hierarchy collapses). While a few loopholes remain, the combination of these sets of results forms a serious hurdle for attempts to bridge average-case and worst-case complexities, and suggest that very new techniques would be required to prove the implication.

Moreover, we rule out "black-box" proofs of some weaker implications. Worst-case to average-case reductions for $L \in NP$ have been shown to put $L$ in a prob-

* Work supported by the Ellentuck Fund, Friends of the Institute for Advanced Study, and NSF grants DMS-0835373 and CCF-o832797 subcontract No. 00001583. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the NSF, Ellentuck Fund, or Friends of the IAS.

1093-0159/11 $26.00 © 2011 IEEE
DOI 10.1109/CCC.2011.34

abilistic and/or non-uniform analog of $co - NP$. Thus, it could be hoped that at least we could give worst-case to average-case reductions for generic languages in $NP \cap co - NP$, showing that $NP \cap co - NP$ has worst-case hard problems if and only if it also has average-case hard problems. Since the worst-case hard problem for our oracle is also in $NP \cap co - NP$ (and even $UP \cap co - UP$), this shows that such an argument could not relativize. Another question is whether average-case easiness for $NP$ implies the same for the polynomial-hierarchy, as it does for worst-case easiness. There was no reason to suspect that such an average-case analog of the worst-case situation would be even non-trivial, but we show a relativized counter-example. (Note also that non-existence of reductions is irrelevant to this issue, since the worst-case result cannot be proved via reductions without collapsing the hierarchy to the second level.)

### *1.1. History and related work*

Average-case analysis of algorithms began in the 1970's, as a way to explain the success of heuristics in solving typical instances of worst-case intractable problems (See e.g., [Karp77]). At the same time, average-case intractability became a desirable property for the fledgling area of modern cryptography. Levin ([Levin86]; see also [G93], [I95], [BT06] for expositions of Levin's theory) introduced a strong and robust theory of average-case complexity, where it was possible to consider the general question of which pairs of problems and input distributions are indeed intractable. He showed the existence of average-case complete distributional problems for $DistNP$.

The concept of *random-self-reducibility* emerged from the need to relate worst-case and average-case hardness for cryptographic problems such as discrete logarithms ([**?**], [**?**], [AFK89], [TW87]). For classes above $NP$, random-self-reducibility ([BF90], [Lip91]) was used to prove the equivalence of worst-case and average-case versions of complete problems. For example, if $PSPACE \neq BPP$, then there are problems in $PSPACE$ that are hard with high probability under the uniform distribution. This results have had important applications to interactive proofs and derandomization ([LFKN92], [BFNW93], [IW97]). However, they do not address the original motivation of putting cryptography on a firmer foundation, by showing that $NP$ problems are $NP$-hard to solve on average.

So many problems above $NP$ are random self-reducible, as are some seemingly hard problems in $NP$ that seem unlikely to be $NP$-complete. Could an $NP$-complete problem be random-self-reducible? Unfortunately, many negative results indicate that random-self-reduction technique cannot apply to $NP$-complete problems ([FF93], [BT03], [AGGM06]). These results consider languages $L \in NP$ so that the worst-case complexity of $L$ reduces to the average-case complexity of a problem of a certain type, via a specified class of reductions. They then show that $L$ is in some sub-class of $NP$, and hence unlikely to be $NP$-complete. These show that the same techniques that proved average-case/worst-case equivalence for the higher classes will fail for $NP$.

However, these results do not completely rule out hope of basing average-case complexity within $NP$ on a worst-case complexity assumption. First, the class of reductions they apply to is limited, and do not include fully adaptive reductions. Secondly, there could be indirect arguments that use the worst-case hardness assumption to prove average-case hardness without giving a reduction. (An analogous example is the worst-case hardness of $NP$ problems following from the worst-case hardness of problems in the higher levels of the polynomial-hierarchy. While this is a simple, relativizing observation, there cannot be a direct reduction unless the hierarchy collapses. ) Thirdly, such results do not rule out basing average-case complexity of problems in $NP$ on a stronger worst-case assumption, such as $NP \cap Co - NP \not\subseteq P/poly$.

We use a standard tool for showing that issues in complexity are non-trivial, *relativization*. In the mid-1970's, Baker, Gill and Solovay [BGS75] introduced relativization as a tool to argue that techniques like simulation and diagonalization cannot, by themselves, resolve the "$P$ vs. $NP$" question. A technique relativizes if it only proves results that also hold when all classes are given the same oracle. If there are oracles giving a contradictory resolution of a complexity question (e.g., $P^A = NP^A$, but $P^B \neq NP^B$), then no relativizing technique can resolve this question. Relativization has been brought to bear upon many other open questions in Complexity Theory. While some non-relativizing techniques were known even before [BGS75], and many recent results especially involving interactive proofs do not relativize, it is still the case that the vast majority of results in complexity theory relativize. The working complexity theorist has a good sense of when a proof outline will relativize, so oracle results remain an important tool to prune away doomed attempts at proofs.

In particular, all of the known worst-case to average-case hardness results for complexity classes relativize.

They all take an arbitrary Boolean function $f$ and construct from it another Boolean function $g = G(f)$ so that any algorithm that computes $g$ on some large fraction of inputs can be used to compute $f$ with high probability on every input. For the same construction to work with an arbitrary $f$, and black-box access to any function correlating with $G(f)$, $G(f)$ must be an error-correcting code for message $f$, since we can recover $f$ from any noisy version of $G(f)$. So the known constructions are all based on locally decodable error-correcting codes. We also need $g$ to be in a small complexity class if $f$ is, so we need $G$ to be computable in a corresponding low-level complexity class. These translated low-level complexity class for classes such as $\#P$ or even $\oplus P$ allow modular counting, and hence versions of Reed-Solomon codes. However, for the polynomial hierarchy, the corresponding class is quasi-polynomial sized constant depth circuits, which cannot compute any function with high average sensitivity, and hence cannot compute any error-correcting code. So the standard approach for worst-case/average-case reductions was already known not to be possible within $NP$ or even within the polynomial-time hierarchy. However, you could easily imagine arguments that used additional functions to define the hard function, i.e, that constructed $g$ not from $f$ itself but from a witness verifier for $f$. Our result rules out most such extensions.

Our paper in some sense answers a "15-year open problem" which we asked in [I95], by showing an oracle where $NP \neq RP$ but $DistNP \subseteq AvgP$. However, in the same paper, we also claimed in passing that there is an oracle relative to which $NP \neq RP$ but $DistNP \subseteq HeurP$, citing an unpublished result of ourselves and Rudich. This claimed result turned out to be fallacious, as we realized recently when attempting to explain it. This false claim probably reduced interest in the problem. However, the current result is substantially stronger than even this retracted claim.

Very recently, Watson [Wat10] has proved that there is an oracle relative to which no worst-case to average-case reductions exist for $NP$-complete languages. Since the combination of such a reduction and $NP \neq RP$ would imply the existence of average-case hard problems in $NP$, these reductions also do not exist relative to the oracles here. So this result implies Watson's main claim. However, he also shows related oracle constructions that seem incomparable to our work. For example, he gives relativizations where there are no reductions from error-free average-case to error-prone average-case heuristics.

### *1.2. Definitions and statement of results*

We follow the definitions and notation from [BT06]. [I95], [BT06] show that these are equivalent to those of [Levin86].

A *distribution ensemble* $D_1, ..D_n...$ is a sequence of distributions, where $D_n$ has support in the $n$ bit strings. A distribution ensemble is *polynomial-time samplable* if there is a polynomial-time algorithm $A$ and a polynomial $poly$ so that $D_n$ is identically distributed to $A(r)$ where $r \in_U \{0,1\}^{poly(n)}$. The *uniform ensemble* is the particular example where $D_n$ is the uniform distribution over all $n$ bit strings.

A *distributional problem* is a pair $(\Pi, D)$ where $\Pi$ is a computational problem and $D$ is a distribution ensemble. The class $DistNP$ is the set of distributional problems where $\Pi$ is the decision problem for a language in $NP$ and $D$ is polynomially samplable.

An *errorless heuristic* $A$ for a problem $\Pi$ is an algorithm that, on input $x$, either outputs a correct solution for $\Pi$ on $x$ or a special error symbol ?. The *error probability* of $A$ on ensemble $D$ is the sequence $error_n = Prob_{x \in_{D_n} \{0,1\}^n}[A(x) =?]$, where the probability is also over the choice of $A$'s random input if $A$ is probabilistic. $AvgP$, respectively, $AvgZPP$ and $AvgP/poly$, are the classes of computational problems $\Pi, D$ so that for every polynomial $p$, there is a deterministic polynomial-time errorless heuristic $A$ (resp., probabilistic polynomial-time errorless heuristic and polynomial-sized circuit family for an errorless heuristic) with the error probability of $A$ on $D$ satisfying $error_n < 1/p(n)$.

A *heuristic* $A$ for a problem $\Pi$ is any algorithm, viewed as attempting to solve $\Pi$. The error-probability $error_n$ for $A$ on $D$ is the sequence $error_n = Prob_{x \in_{D_n} \{0,1\}^n}[A(x)$ is not a correct answer to $\Pi$ on $x]$, where the probability is also taken over the random coin flips of $A$ if any. $HeurP$, respectively, $HeurBPP$ and $HeurP/poly$, are the classes of computational problems $\Pi, D$ so that for every polynomial $p$, there is a deterministic polynomial-time heuristic $A$ (resp., probabilistic polynomial-time heuristic and polynomial-sized circuit family for a heuristic) with the error probability of $A$ on $D$ satisfying $error_n < 1/p(n)$.

Note that, when we relativize $P$ and $BPP$ to an oracle $O$, this induces relativized definitions of all average-case classes above, since they are defined in terms of polynomial-time computation. Let $SIZE^O(T(n))$ denote the class of functions with circuit families with gates for oracle $O$ of size $O(T(n))$.

Our main theorem is then:

**Theorem 1.** *There is an $\alpha > 0$ and an oracle $O$ so*

*that* $DistNP^O \subseteq AvgP^O$ *and* $UP^O \cap co - UP^O \not\subseteq SIZE^O(2^{n^\alpha})$.

**Corollary 2.** *There is an oracle* $O$ *with* $DistNP^O \subseteq AvgP^O$ *and* $NP^O \neq RP^O$.

We also show:

**Theorem 3.** *There is an* $\alpha > 0$ *and an oracle* $O$ *so that* $DistNP^O \subseteq AvgP^O$ *and* $\Sigma_2^O \not\subseteq HeurSIZE^O(2^{n^\alpha})$.

The corollary follows from either theorem.

### *1.3. Intuitive overview*

One complication for our oracle construction is a (relativizing) result of Gutfreund, Shaltiel, and Ta Shma [GST05]. We can think of average-case complexity as competition between a Challenger, creating an instance, and a Solver, trying to find a solution, where both are time limited ([G93]). We want to show an oracle relative to which, if the Challenger spends time $T$ to create an instance, the Solver usually can solve it in time $poly(T)$. So we want the Solver not to spend much more time than the Challenger. On the other hand, [GST05] show that if $NP \not\subseteq BPP$, then for any fixed algorithm $A$ running in polynomial time $T$ for an $NP$-complete problem, there is a distribution samplable in time $poly(T)$ that finds instances where the algorithm fails. That is, if $NP$ problems are hard in the worst-case, the Solver must spend at least an amount of time polynomial in the Challenger's time. (By the reduction of [IL90], this issue arises even if we restrict our attention to uniform distributions; then, time to generate instances will correspond to rarity under the uniform distribution.)

In other words, we cannot just divide instances into "easy" and "hard", with almost all being "easy"; we need a gradual tradeoff where a smaller and smaller number of instances become harder and harder.

Most of the oracle constructions one might think of (and all of the ones we tried, including this one) would have the form $F + A$, where $F$ is intended to create a hard problem in $NP$, and $A$ would provide advice to make most instances of problems in $NP^F$ easy. The problem is that the Challenger also has access to $A$, and the problem of generating instances where oracle $A$ fails is in $NP^{F+A}$. To handle the need for a rarity/hardness tradeoff, we build a stratification into our oracle construction. Our basic "hard problem" will be inverting a random permutation. This type of hard problem has the feature that it is easy to identify sub-problems of the same type: invert the function on a subset of the range (and hence, find inverses in a subset of the domain). The stratification into distributions $D_0, D_1, ..D_i..$ of harder and harder instances will be the permutation restricted to random subsets of the range and domain of decreasing size. To make $NP$ problems easy on average, we'll construct a stratified oracle $A$ that answers $NP$ type questions, but censors answers based on information about the higher levels of this stratification of instances. The longer the query made to $A$, the deeper the level of the stratification where censorship occurs.

We make essential use of permutation and matching restrictions from proof complexity. Permutation restrictions have a clear notion of "easy" restricted instances (ones determined by the restriction) and a clear class of hard restricted instances (invert the permutation where this inverse is not determined by the restriction.) Other kinds of restriction would give the same notion of "easy" restricted instance, but for, e.g., random restrictions, it is not clear how to define any class of "hard" instances that depend in an essential way on the unrestricted part. As a side benefit, the worst-case hard problem will be in $UP^O \cap co - UP^O$. To prove theorem 3, we instead use random 1-1 functions to a range slightly larger than the domain.

## 2. Random restrictions of matching variables

Our construction uses results about random restrictions of Boolean variables representing matchings or partial matchings. Switching lemmas for this type of variables were originally introduced to prove lower bounds on the proof complexity of the pigeon-hole principle ( [KPW95], [PBI93], [R93], [B94]). The switching lemmas basically say that randomly restricting a small fan-in DNF formula over matching variables with high probability makes it equivalent to a small depth matching decision tree. Here, we are interested in the degenerate case of switching lemma where the tree has depth 0, i.e., the formula becomes constant. Since the previous work was not tailored to this degenerate case, and for completeness, we give a self-contained proof of the needed bound here. However, we are essentially following the counting argument from [R93], [B94].

Let $D$ and $R$ be a finite domain and range for a matching, both of size $N$. The *matching variables* are $x_{i,j}, i \in D, j \in R$, intuitively saying that $i$ is mapped to $j$ by a permutation from $D$ to $R$. A *$T$-matching term* is the conjunction of at most $T$ matching variables that form a partial matching, i.e., no two variables have the same value of $i$ or of $j$. A *$T$-matching DNF* is a disjunction of $T$-matching terms. Let $D' \subseteq D, R' \subseteq R$, $|D'| = |R'| = K$, and let $\sigma$ be a matching from $D - D'$ to $R - R'$. The *restriction* $\rho$ corresponding to $D, R, D', R', \sigma$ assigns variable $x_{i,j}$ value 1 if $i \in$

$D - D', j \in R - R'$ and $\sigma(i) = j$, undefined if $i \in D'$ and $j \in R'$, and 0 otherwise. A random restriction for $D, R, K$ is one obtained by picking $D'$ and $R'$ of size $K$ independently and uniformly at random, then picking $\sigma$ uniformly at random among all permutations from $D - D'$ to $R - R'$. Let $\Phi$ be a $T$-matching $DNF$ and let $\rho$ be a restriction as above. We say that $\Phi|_\rho$ is 1 if all the variables in some term of $\Phi$ are set to 1 by $\rho$, and $\Phi|_\rho = 0$ if all terms contain a variable set to 0. $\Phi|_\rho$ is *constant* if it is either 1 or 0. Equivalently, $\Phi|_\rho$ is constant if it is constant on all matchings consistent with $\rho$.

**Lemma 4.** *Let $1 \leq T < K$. Let $|D| = |R| = N$, and let $\Phi$ be a $T$-matching $DNF$ over the corresponding variables. Let $\rho$ be a random $(D, R, K)$ restriction. Then the probability that $\Phi|_\rho$ is not constant is at most $8TK^2/N$.*

**Proof:** If $8TK^2 \geq N$, the claim is trivial, so we assume $8TK^2 < N$ so in particular, $N > 8K$.

We will show how to "code" every $\rho$ so that $\Phi|_\rho$ is not constant, in a way that $\rho$ is recoverable from its code. Then we'll bound the number of codes.

The code will consist of an integer $L > 0$, a $(D, R, K-L)$ restriction $\rho'$ extending $\rho$, and a subset $A$ of positions in $1, ..T$ of size $L$. Since $\Phi|_\rho$ is not constant, there is a first term $t$ in it so that no variable in $t$ is set to 0. Also, not all variables in any term are set to 1. Let $T \geq L \geq 1$ be the number of unassigned variables in $t$, and let $\rho'$ extend $\rho$ by setting these $L$ variables to 1, i.e., extending $\sigma$ to include the partial matching on these variables, and subtracting their first co-ordinates from $D'$ and their second from $R'$. Let $A$ be the set of positions in the term these variables occupy.

Given $L, \rho'$, and $A$, we decode $\rho$ as follows. We take the first term in $\Phi$ that is entirely set to 1 by $\rho'$. This is $t$, since all earlier terms had one variable set to 0 in $\rho$. Then we unassign the variables in positions in $t$ corresponding to $A$, and add their first and second co-ordinates to $D'$ and $R'$ respectively.

The number of $(D, R, K)$ restrictions is $\binom{N}{K}^2 (N - K)!$. The number of sets $A$ of size $L$ is at most $T^L$. Thus, the fraction of $\rho$ with such a code is at most $\sum_{L=1}^{L=T} (T^L \binom{N}{K-L}^2 (N - K + L)! / \binom{N}{K}^2 (N - K)! \leq \sum_{L=1}^{L=\infty} (TK^2 N/(N-K)^2)^L$. Since $N > 8K$, this is at most $\sum_L (4TK^2/N)^L$. Since by assumption, $4TK^2/N \leq 1/2$, the geometric sum is bounded by twice the first term, $8TK^2N$. **QED**

We can extend this to variables that come from matchings on several different domains and ranges, and where the domains and ranges are not of identical size. Let $D_1, ...D_m$ and $R_1, ..R_m$ be disjoint sets of size $N_1, ..N_m$ and $N_1 + \Delta_1 ... N_m + \Delta_m$ respectively. Consider the variables $x_{i,j}$, where there is a $k$ with $i \in D_k$ and $j \in R_k$. A $T$-matchings term is a conjunction of at most $T$ such variables so that within each $D_k, R_k$, the variables in the term form a partial matching. A $T$-matchings DNF is a disjunction of $T$-matchings terms. Let $K_1, ..K_m$ be integers, $1 \leq K_k \leq N_k$. Pick corresponding random restriction $\rho_1, ...\rho_k$ for each triple $D_k, R_k, K_k$ and let $\rho$ be the union of such restrictions (note that here $R'_i$ will be of size $K_i + \Delta_i$ rather than $K_i$.)

**Lemma 5.** *Let $T, D_1, ..D_m, R_1, ..R_m$ , $N_1, ..N_m, \Delta_1, ..\Delta_m, K_1, .K_m$ be as above. Assume each $\Delta_i \leq K_i$, and let $r = max_k K_k^2/N_k$. Let $\Phi$ be a $T$-matchings $DNF$ over the corresponding variables. Let $\rho$ be a random restriction picked as described above. Then the probability that $\Phi|_\rho$ is not constant is at most $8mrT$.*

**Proof:** As before, we can assume that $8mrT < 1$, and so in particular, we will have $N_k > 8K_k$ for all $k$. As before, we will show how to "code" every $\rho$ so that $\Phi|_\rho$ is not constant, in a way that $\rho$ is recoverable from its code. Then we'll bound the number of codes.

The code consists of integers $L_1, ..L_m$, a $(D_1, ..D_m, R_1, ..R_m, K_1 - L_1, ....K_m - L_m)$ restriction $\rho'$ extending $\rho$, and a subset $A$ of positions in $1, ..T$ of size $L = L_1 + L_2 + .. + L_m \geq 1$. Since $\Phi|_\rho$ is not constant, there is a first term $t$ in it so that no variable in $t$ is set to 0. Also, not all variables in any term are set to 1. Let $T \geq L_i \geq 0$ be the number of unassigned variables in $t$ from $D_i, R_i$, and let $\rho'$ extend $\rho$ by setting these $L = \sum L_i \geq 1$ variables to 1. Let $A$ be the set of positions in the term these variables occupy.

Given $\rho'$ ( which determines $\vec{L}$) and $A$, we decode $\rho$ as follows. We take the first term in $\Phi$ that is entirely set to 1 by $\rho'$. This is $t$, since all earlier terms had one variable set to 0 in $\rho$. Then we unassign the variables in positions in $t$ corresponding to $A$, and add their first and second co-ordinates to the relevant $D'_k$ and $R'_k$ respectively.

For a fixed $L_1, , ..L_m$, with $L_k > 0$, we count the number of possible $\rho'$ as a fraction of the number of possible $\rho$. The number of $(D_i, R_i, K_i)$ restrictions is $\binom{N_i}{K_i}\binom{N_i+\Delta_i}{K_i+\Delta_i}(N_i - K_i)!$. The same calculation as before shows that the number of $(D_i, R_i, K_i - L_i)$ restrictions is at most a $(K_i(K_i + \Delta_i)N_i/(N_i - K_i)(N_i + \Delta_i - K_i))^{L_i}$ fraction of the $(D_i, R_i, K_i)$ restrictions. Since $N_i > 8K_i > 8\Delta_i$, this is at most $(4K_i^2/N_i)^{L_i} \leq (4r)^{L_i}$, so the to-

tal number of $(D_1...D_m, R_1, ..R_m, K_1 - L_1, ..K_m - L_m)$ restrictions as a fraction of the number of $(D_1, ..D_m, R_1, ..R_m, K_1, ..K_m)$ restrictions is at most $(4r)^L$. To get the total fraction of $\rho$ with a code, we also need to multiply by the number of ways of partitioning $L$ into $L_1, ..L_m$ which is at most $m^L$, and the total number of sets $A$, which is at most $T^L$. This gives us the total fraction of non-constant $\rho$ is at most $\sum_L (4rmT)^L \leq 8rmT$. **QED**

## 3. The oracle construction, simplified

In this section, we present the main idea of the oracle construction, but with two simplifications that we eliminate later. First, we will think of the oracle as a finite object, only defined for query lengths between $n$ and $2^n$, and look at the behavior of computation for sufficiently large $n$. Later, we will have to argue that the parts of the construction for the different lengths $n$'s don't interfere with each other. Second, we only show that problems are easy on average for the uniform distribution. Later, we'll show that this is true for any samplable distribution. [IL90] and [BT03] show that for *randomized* algorithms, average-case easiness on samplable distributions follows from that on the uniform distribution. However, we will argue directly that a determininstic algorithm works on any samplable distribution.

Let $n$ be a large integer, and let $D_0 = R_0 = \{0,1\}^n$. Let $max = (1/8)\log n$, and let $N_0 = 2^n$, and $N_i = 2^{n/8^i}$ for $i$ from 1 to $max$. Pick $F$ as a random permutation from $D_0$ to $R_0$, and pick $D_{max} \subset D_{max-1} ... \subset D_1 \subset D_0$, at random so that $|D_i| = N_i$ for each $i$. Let $R_i = F(D_i)$. Let $\rho_i$ be the restriction consistent with $F$ and undefined on pairs in $D_i \times R_i$. For notational uniformity, we define $\rho_{max+1} = F$.

One part of our oracle will be $F$, and our worst-case hard problem will be to invert $F$. Our oracle will be of the form $F + A$ for an oracle $A$ intended to make $NP$ problems easy on average, but censor information about $D_i$ for large $i$. $A$ is defined as follows:

On input $(M, x, 1^{T^4})$ where $M$ is the description of a non-deterministic oracle machine, perform the following procedure:

If $T > 2^{n^{1/4}}$, then return a bit saying whether $M$ has a time $T$ accepting path with oracle $F + A$ accepting $x$.

Otherwise, let $i = 1/2 \log\log T$. For each $T$-time accepting computation path for $M^{F'+A}(x)$ where $F'$ is any permutation from $D_0$ to $R_0$, write down a term containing all variables $x_{q,F'(q)}$ where $M$ makes query $q$ to $F'$. Let $\Phi_0$ be the resulting $T$-matching DNF, and let $\Phi_i$ be $\Phi$ restricted by $\rho_i$. If $\Phi_i$ is constant, return that constant. Otherwise, return ?.

Note that this procedure is recursive, in that we simulate $M$ with the actual answers to queries in $A$, not potential answers as we do for queries to $F$. However, it is not circular, since in a query to $A$ of length $L$, $T \leq L^{1/4}$, so all queries made when simulating $M$ for $T$ steps will be of length $L' \leq L^{1/4} < L$.

**Lemma 6.** *Answers to queries to $A$ of the form $(M, x, 1^{T^4})$ are determined by $\rho_1, ...\rho_i$, where $i = 1/2 \log\log T$.*

**Proof:** By induction on $T$. All the queries $M$ could ask $A$ in the simulation constructing $\Phi_0$ are of the form $(M', x, 1^{(T')^4})$ where the total length is less than $T$, so $T' < T^{1/4}$. So by induction, the answers to all of these queries are determined by $\rho_1, ...\rho_{i-1}$, since $1/2 \log\log T' \leq 1/2 \log\log T - 1 = i - 1$. So this information determines $\Phi_0$. Then the answer to the query is determined by $\Phi|_{\rho_i}$, which is determined by $\Phi_0$ and $\rho_i$.. (Note that, if $T > 2^{n^{1/4}}$, $i > max$, and the information we claim determines the answer involves all random variables used to construct the oracle, so the claim is trivial.) **QED**

**Lemma 7.** *With high probability, any circuit to invert $F$ on inputs of size $n$ with oracles for $F, A$ requires size $\Omega(2^{n^{1/4}})$.*

**Proof:** Consider any potential circuit of size less than $2^{n^{1/4}}$.

All queries such a circuit could make to $A$ have $T < 2^{n^{1/4}}$, so are determined by $\rho_1, ..\rho_j$ for $j = 1/2 \log\log T < 1/8 \log n = max$. In particular, such queries are independent of $F$ restricted to $D_{max}$, so even after fixing the answers to all such queries, this restricted function is a random map between two sets of size $N_{max} = 2^{n/8^{1/8 \log n}} = 2^{n^{5/8}}$.

Fixing all queries to $F$ outside $D_{max}$ leaves a circuit that makes queries to a random matching from $D_{i_{max}}$ to $R_{i_{max}}$ to find the inverse for an element of $R_{i_{max}}$.

Information-theoretically, this is equivalent to inverting a random permutation on a set of the same size. Several researchers have shown that this is hard by different techniques and with different quantitative trade-offs [Yao90], [I96], [Zim], [GT00], [DTT10]. In particular, the (unpublished) counting arguments from [I96], [Zim], imply that such a map requires $2^{\Omega(n^{5/8})}$ circuit size relative to $F$. For completeness, we include the full argument in the appendix (Theorem **??**) using this approach. However, we could use any of the other published methods as an alternative. **QED**

So relative to this oracle, there are worst-case hard problems. We need to show that all problems in $NP$ are easy on most instances.

**Lemma 8.** *For any $M, x, T$, the probability that $A$ returns ? on $(M, x, 1^{T^4})$ is at most $2^{-\Omega(n^{5/8})}$.*

**Proof:**

If $T \geq 2^{n^{1/4}}$, then $A$ never returns ?. If not, let $i = 1/2 \log\log T < max$. All queries $M^{F'+A}(x)$ makes to $A$ on $T$ step paths are of the form $(M', x, 1^{(T')^4})$ where $T' \leq T^{1/4}$. so are determined by $\rho_1, ..\rho_{i-1}$. Let $\Phi$ be the $T$-matching $DNF$ described in the construction of $A$, and consider $\Phi' = \Phi|_{\rho_{i-1}}$

We can think of the new part of $\rho_i$ as a random restriction chosen on universe $D_{i-1}, R_{i-1}$ leaving $N_i$ variables unset. By lemma 4, the probability that $\Phi'|_{\rho_i} = \Phi_{\rho_i}$ is not constant is at most $8T(N_i)^2/N_{i-1} = 8T2^{2n/8^i - n/8^{i-1}} = 8T2^{-6n/8^i} \leq 82^{n^{1/4}} 2^{-.75n^{5/8}} \in 2^{-\Omega(n^{5/8})}$. **QED**

**Corollary 9.** *Let $M$ be any non-deterministic machine that runs in time $T(n)$. With probability $1 - 2^{-\Omega(\sqrt{n})}$ over random oracles $F + A$, there is a deterministic algorithm running in time $O(T(n)^4)$ that never returns a Boolean value distinct from $L_M(x)$, and returns ? with probability for at most a $2^{-\Omega(\sqrt{n})}$ fraction of inputs $x$ of length $n$.*

**Proof:** The algorithm on input $x$, merely queries $A$ at $(M, x, 1^{(T(n))^4})$. By the construction of $A$, when it returns a Boolean value, this is identical to $M^{F+A}(x)$. By the previous lemma, for any fixed $x$, the probability of failing to return a value is at most $2^{-\Omega(n^{5/8})}$, so the probability bound in the claim follows from a simple averaging argument. **QED**

If our goal were to get a problem that is hard for infinitely many lengths $n$, we could use the above argument with a sparse sequence of lengths $n_0 << n_1 << n_2 << n_3$.... However, in the next section, we will show how to use the same idea to get a problem that is hard for all input sizes.

## 4. THE FULL CONSTRUCTION

Here, we modify the above construction to handle hard problems for every input size, and show that $NP$ becomes easy on any samplable distribution, not just the uniform distribution.

For each integer $n$ let $D_{n,0} = R_{n,0} = \{0,1\}^n$. Let $N_{n,0} = 2^n$, and $N_{n,i} = 2^{n/8^i}$ for $i$ from 1 to $max_n = (1/8)logn$. Pick $F_n$ as a random permutation from $D_{n,0}$ to $R_{n,0}$, and pick $D_{n,i_{max}} \subset D_{n,i_{max}-1} \subset ...D_{n,1} \subset D_{n,0}$, at random so that $|D_{n,i}| = N_{n,i}$. Let $R_{n,i} = F_n(D_{n,i})$. Let $\rho_{n,i}$ be the restriction corresponding to $D_{n,0}, R_{n,0}, D_{n,i}, R_{n,i}, F_n$. For notational convenience, we define $\rho_{n,i} = F_n$ for $i > max_n$.

One part of our oracle will be $F = \cup_n F_n$, and our wost-case hard problem will be to invert $F_n$. The oracle will be of the form $F + A$, where, as before, $A$ is intended to make $NP$ problems easy on average, but censor information about $F_n$ on $D_{n,i}$ for large $i$. $A$ is defined as follows:

On input $(M, x, 1^{T^4})$ where $M$ is the description of a non-deterministic oracle machine, perform the following procedure:

Let $i = 1/2 \log\log T$. For each $T$-time accepting computation path for $M^{F'+A}(x)$ where each $F'_n$ is a permutation from $D_{0,n}$ to $R_{0,n}$, write down a term containing all variables $x_{q,F'(q)}$ where $M$ makes query $q$ to $F'$. Let $\Phi$ be the resulting $T$-matchings DNF. Note that $M$ can never make a query longer than $T$, so $\Phi$ will involve matching variables in at most $m = T$ pairs $(D_{1,0}, R_{1,0})...(D_{T,0}, R_{T,0})$.

Restrict $\Phi$ by $\rho_{1,i}, ..\rho_{T,i}$. Note that this restriction will always restrict variables in $D_{n,0}, R_{n,0}$ where $T > 2^{n^{1/4}}$, since then $i > max_n$. If the result is constant, return that constant. Otherwise, return "?".

As before, this procedure is recursive, in that we simulate $M$ with the actual answers to smaller queries in $A$, not potential answers as we do for queries to $F'$. As before, all simulated queries will be of length $L' < T$, so will involve times $T' \leq T^{1/4}$.

**Lemma 10.** *The answers to query $(M, x, 1^{T^4})$ to $A$ is determined by the set of $\rho_{n,j}$ where $n \leq T$ and where $j \leq 1/2 \log\log T$.*

**Proof:** By induction on $T$. As before, all the queries $M$ could ask $A$ on any path in the simulation are inductively determined by the above information, with $j \leq i - 1$, which determines $\Phi$. Then the answer to the query is determined by $\Phi|_{\cup_n \rho_{n,i}}$. for $i = 1/2 \log\log T$ and the union is over $n \leq T$. **QED**

The main theorem then follows from the following two lemmas:

**Lemma 11.** *With high probability, any circuit to invert $F$ on inputs of size $n$ with oracles for $F, A$ requires size $\Omega(2^{n^{1/4}})$.*

**Proof:** Consider any potential circuit with oracle $F + A$ for inverting $F$ on inputs of length $n$. Fix all the functions $F_{n'}$ and restrictions $\rho_{n',j}$ for $n' \neq n$.

All queries to $A$ such a circuit could make have $T < 2^{n^{1/4}}$, so are determined by $\rho_{n,j}$ for $j \leq 1/2 \log\log T < 1/8 \log n = max_n$. In particular, such

queries are independent of $F_n$ restricted to $D_{max_n}$, so even after fixing the answers to all such queries, this restricted function is a random map between two sets of size $|D_{max_n}| = 2^{n/8^{1/8 \log n}} \geq 2^{n^{5/8}}$. Information-theoretically, this is equivalent to inverting a random permutation on a set of the same size, so Theorem **??** implies that such a map requires $2^{\Omega\sqrt{n}}$ circuit size relative to $F$. **QED**.

So relative to this oracle, there are worst-case hard problems. We need to show that all problems in $NP$ are easy on most instances, on all samplable distributions.

**Lemma 12.** *Let $M^{F+A}$ be a non-deterministic polynomial-time oracle machine. Let $S^{F+A}$ be a probabilistic polynomial-time oracle sampling algorithm using a random string of length $l(n)$ to output a string of length $n$. Then with probability 1 over choice of oracles $F$ and resulting $A$, there is a deterministic polynomial-time errorless heuristic using oracle $F+A$, $B^{F+A}$, with $error_n = n^{-\omega(1)}$.*

**Proof** The algorithm $B$ is simple. On input $x$ of length $n$, let $T$ be the maximum of $n$, the fourth power of the time $S$ takes on inputs of length $l(n)$ and the time $M$ takes on inputs of size $n$. Query $A$ at $(M, x, 1^{T^4})$ and return the answer.

By the construction of $A$, if the answer is Boolean, it is equal to $M^{A+F}(x)$. We need to bound the probability that it is "?".

Fix a random input $r$ to $S$. Let $T$ be as above, and let $i = 1/2 \log\log T$.

Note that the instance $x = S^{F+A}(r)$ is a random variable itself. We first show that with high probability, $x$ is determined by the restrictions $\rho_{n,j}$ with $n \leq T^{1/4}$ and $j \leq i-1$.

Because all queries $S$ makes to $A$ are of length at most $T^{1/4}$, and hence involve $T' < T^{1/16}$, they are determined by the restrictions $\rho_{n,j}$ with $n \leq T^{1/16}$ and $j \leq i-2$. Fix all such restrictions and hence answers to all queries $S$ makes to $A$. Then $S^{F+A}(r)$ can be simulated as a $T^{1/4}$ query decision tree with queries to the $F_n$'s on $D_{n,i-2}$, with $n < T^{1/4}$.

For any such restrictions, the conditional probability that $S^O(r)$ is not determined by $rho_{n,j}$ for $n \leq T^{1/4}$ and $j \leq i-1$ is bounded by the probability that one of the queries $S^{F+A}(r)$ makes is in $D_{n,i-1}$. Note that if $n < \log^4 T$, $i - 1 \geq max_n$, so $D_{n,i-1}$ is empty, i.e, the restriction is $F_n$ itself. Thus, only such queries with $n \geq \log^4 T$ are possible. Thus, the conditional probability that $S$ asks any such query to some $F_n$ that is left unset by $\rho_{n,i-1}$ is at most

$$T^{1/4} max_{T^{1/4} \geq n \geq \log^4 T} N_{n,i-1}/N_{n,i-2}$$

$$\leq T^{1/4} max_{n' \geq \log^4 T} (2^{(7/8)n'/8^{1/4 \log\log T}})$$

$$\leq T^{1/4} 2^{7/8 \log^4 T/\log T} \leq T^{-\omega(1)}$$

.

Assume the above unlikeley event does not occur. Then the restrictions $\rho_{n,j}$ with $j \leq i-1$ determine $x = S^{A+F}(r)$. They also determine all queries $M$ might ask $A$ in time $T$, and hence the value of $\Phi$ constructed in the procedure for the oracle query $(M, x, 1^{T^4})$ $B$ makes to $A$.

Now given these restrictions, we determine from $x$ the $T$-matchings DNF $\Phi$ corresponding to $x$, with variables from $(D_{n,0}, R_{n,0})$ with $1 \leq n \leq T$. Let $\Phi'$ be the restriction of $\Phi$ by $\rho_{n,j}$ for $j < i$, $n \leq T$. The probability that $A$ responds to our query with "?" is the probability that $\Phi'$ , further restricted by the union of $\rho_{n,i}$ for $\log^4 T \leq n \leq T$ i, is non-constant. (As noted above, matching variables with $n < \log^4 T$ have all been restricted already in $\Phi'$.) This probability is bounded by Lemma 5, where we have $m < T$ families of restrictions, each leaving $N_{n,i}$ out of $N_{n,i-1}$ elements unset, and each $\Delta_i = 0$. Therefore, the conditional probability is at most

$8T^2 max_{n' \geq \log^4 T}(N^2_{n',i}/N_{n',i-1}) =$
$8T^2 max_{n' \geq \log^4 T}(2^{2n'/8^i - n'/8^{i-1}}) =$
$8T^2 max_{n' \geq \log^4 T} 2^{-6n'/8^{1/2 \log\log T}} \leq$
$8T^2 2^{-\Omega(\log^4 T/\log^{3/2} T)} = T^{-\omega(1)}$.

Thus, the overall probability that the $B$ returns ? is at most the probability that $x$ is not determined by the restrictions, plus the probability that $\Phi$ is not made constant, both of which are $T^{-\omega(1)} \leq n^{-\omega(1)}$, so less than any polynomial. **QED**

Now the above probability is over the randomness in the oracle construction, as well as the internal randomness of the sampler, but by Markov's inequality, the probability over $F, A$ that the conditional failure probability for $B$ is greater than $n^2$ times its expectation is at most $1/n^2$. Since $\sum 1/n^2$ converges, with probability 1, there are only finitely many $n$ where it fails. So with probability 1, the failure probability for $B$ is asymptotically at most $n^2 n^{-\omega(1)} = n^{-\omega(1)}$.

This proves Theorem 1. **QED**

## 5. No average-case collapse of polynomial hierarchy

Here, we modify the above argument to show that we can have all problems in $NP^O$ be easy on average, but hard average-case problems in $\Sigma_2^O$. Thus, the counter-part to the collapse of the polynomial-time hierarchy might fail with respect to average case complexity. Such oracles also separate average-case and

worst-case complexities of $NP$. The change in the basic construction is that instead of choosing a random permutation $F_n$ from $\{0,1\}^n$ to $\{0,1\}^n$, we choose random 1-1 function between the same domain and a range somewhat larger. The average-case hard problem in $\Sigma_2^O$ will be to determine whether there is an element in an interval that is not in the image of $F_n$.

For each integer $n$ let $\Delta_n = 2^{n^{1/8}}$, let $D_{n,0}$ be of size $2^n$ and let $R_{n,0}$ be of size $2^n + \Delta_n$. Let $N_{n,0} = 2^n$, and $N_{n,i} = 2^{n/8^i}$ for $i$ from 1 to $max_n = 1/8logn$. Pick $F_n$ as a random 1-1 function from $D_{n,0}$ into $R_{n,0}$, and pick $D_{n,max_n} \subset D_{n,max_n-1} \subset ...D_{n,1} \subset D_{n,0}$, at random so that $|D_{n,i}| = N_{n,i}$. Let $R_{n,i} = F_n(D_{n,i})$. Let $\rho_{n,i}$ be the restriction corresponding to $D_{n,0}, R_{n,0}, D_{n,i}, R_{n,i}, F_n$. We define $\rho_{n,j} = F_n$ for $j > max_n$.

As before our oracle will be of the form $F + A$ for an oracle $A$ designed to make $NP$ problems easy on average, but censor information about $D_{n,i}$ for large $i$. $A$ is defined as follows, almost the same as before:

On input $(M, x, 1^{T^4})$ where $M$ is the description of a non-deterministic oracle machine, perform the following procedure:

Let $i = 1/2 \log\log T$. For each $T$-time accepting computation path for $M^{F'+A}(x)$ where each $F'_n$ is a 1-1 function from $D_{0,n}$ to $R_{0,n}$, write down a term containing all variables $x_{q,F'(q)}$ where $M$ makes query $q$ to $F'$. Let $\Phi$ be the resulting $T$-matchings DNF.

For each $1 \leq n \leq T$, restrict $\Phi$ by $\rho_{n,i}$. If the result is constant, return that constant. Otherwise, return "?".

As before, this procedure is recursive, in that we simulate $M$ with the actual answers to smaller queries in $A$, not potential answers as we do for queries to $F'$. As before, all simulated queries will be of length $L' < T$, so will involve times $T' \leq T^{1/4}$.

The proof that $DistNP^O \subseteq AvgP^O$ is identical to before. We now show that with high probability, functions in $\Sigma_2^O$ require high circuit complexity even to get a good heuristic performance on average instances.

Partition the set $R_{n,0}$ into $2\Delta_n$ subsets of equal size, arbitrarily but with it being easily decidable which subset an element is in (e.g., do it lexicographically). Let these subsets be called $U_{n,1}, ..U_{n,2\Delta_n}$. Since there are $\Delta_n$ random elements that are not preimages of $F$, we expect a subset to contain one such element with constant probability, a little less than 1/2 since several such could fall in a single subset. Let $L^O$ be the language $\{(n,i) : 1 \leq i \leq 2\Delta_n, \exists y \in U_{n,i} - F_n(D_{n,0}\}$. Since to decide whether a string is not in the range of $F_n$ is in $Co - NP^O$, $L^O$ is always in $\Sigma_2^O$.

**Lemma 13.** *With high probability, any circuit $C_n^O$ which, for more than .9 fraction of $1 \leq i \leq \Delta_n$ answers whether $(n,i) \in L^O$ has size at least $2^{n^{1/16}}$.*

**Proof:** As before, we can condition on the choice of the $F_{n'}$ and $\rho_{n',j}$ for $n' \neq n$ and the choices of all $\rho_{n,i}$ with $i < max_n$ which determine all queries in $A$ of length at most $2^{n^{1/4}}$. Then the remaining unset queries of $F_n$ form a random 1-1 function from a set of size $N_{n,max_n}$ into a set of size $N_{n,max_n} + \Delta_n$. Let $E = (N_{n,i_{max}} + \Delta_n)/2\Delta_n$ be the expected number of elements in the unrestricted part of $R_{n,i_{max}}$ within any fixed $U_{n,i}$. Since the unrestricted set is randomly chosen with high probability each such set has at least $E(1 - o(1))$ such unrestricted elements.

Consider any potential circuit $C$ with oracle $F + A$ for computing whether $(n,i) \in L^O$, that makes at most $Q < 2^{n^{1/16}}$ queries to its oracle. Simulate $C$ on all $2\Delta$ inputs $i$, thus fixing the answers to at most $2Q\Delta_n$ queries to $F_n$. Since $2Q\Delta_n = o(E)$, even with these queries fixed, each set $U_{n,i}$ has $E(1 - o(1))$ unmatched elements, and the outputs for $C^O(i)$ are now completely determined for all $i$. If $C^O(i) = 1$ for $1.2\Delta_n$ values of $i$, any fixing of $F_n$ will cause $.2\Delta_n$ of them to be incorrect, since there are only $\Delta_n$ non-members of the image of $F_n$. Then $C$ is incorrect with probability .1 over choice of $i$. So assume $C^O(i)$ outputs 0 for at least $.8\Delta_n$ inputs $i$. Conditioned on the information previous, the non-members of the image of $F_n$ are a uniformly chosen set of $\Delta_n$ unmatched elements of $R_{n,i_{max}}$. Consider picking them one by one. While there are fewer than $.2\Delta_n$ sets $U_{n,i}$ with $C^O(i) = 0$ that have a chosen element, there are at least $.6\Delta n$ such $i$'s currently without a chosen element, and their union contains at least a .25 fraction of all unmatched, unchosen elements. Thus, until the $.2\Delta_n$ limit on mistakes is reached, there is a .25 chance that the next element will add one to the set of mistakes for $C^O$. Thus, the probability of never reaching the $.2\Delta_n$ limit is at most that of flipping $\Delta_n$ coins of bias .25 and having fewer than .2 fraction of them return heads, which by Chernoff bounds, is at most $exp(-\Theta(\Delta_n))$. Thus, with high probability, there is no such circuit $C$ of size $o(2^{n^{1/16}})$. **QED**

## 6. Open Problems

Several open problems present themselves. Can we show that algebrazing techniques cannot prove worst-case to average-case hardness conversions for $NP$ ([AW08], [IKK])? Can we construct oracles separating average-case complexity from heuristic complexity? A major open problem is whether average-case easiness for $NP$ extends to imply average-case easiness of

problems in the polynomial hierarchy. By using the full switching lemma for matching variables ([KPW95], [PBI93], [B94], [R93]), our techniques should extend to construct oracles where $NP$ problems are hard on the worst-case but all problems in the polynomial hierarchy are hard on average. Can we construct oracles where problems in $NP$ are easy on average, but not problems in the polynomial hierarchy?

The most important open problem is to give complexity-theoretic evidence that some problems in $NP$ are in fact average-case intractable, and more particularly, to base this conclusion on a plausible worst-case intractability assumption. While the combination of our oracle construction and the known limitations on randomized reductions make this task seem difficult, there are still some loop-holes. Oracle results rule out "fully black-box" reductions but not black-box reductions from particular problems ([RTV04]). The difference is that in a fully black-box reduction, the average-case hard problem is constructed in a black-box manner from the worst-case hard problem. On the other hand, [BT03] rules out non-adaptive black-box reductions from any $NP$-complete problem unless the polynomial-time hierarchy collapses. It might be possible to show that a non-fully black-box reduction holds based on any particular problem in $NP \cap co-NP$, or that an adaptive black-box reduction is possible from a particular $NP$-complete problem. Possibly we could show that a complete problem from a subclass of $NP$ such as $PLS$ is random-self-reducible. It is also possible that the implication is provable using a non-relativizing indirect technique, which does not yield any kind of reduction.

## 7. ACKNOWLEDGEMENTS:

We would like to thank Luca Trevisan for prompting us to revisit this problem. We would also like to thank Valentine Kabanets, Paul Beame, and the reviewers for valuable comments.

## REFERENCES

[AFK89] M. Abadi, J. Feigenbaum, and J. Kilian, On Hiding Information from an Oracle, *J. Comput. System Sci.*, 1989

[AGGM06] A. Akavia, O. Goldreich, S. Goldwasser, and D. Moshkovitz, On basing one-way functions on $NP$ completeness, *STOC*, 1986.

[AIV92] S. Arora, R. Impagliazzo, and U. Vazirani. Relativizing versus nonrelativizing techniques: The role of local checkability. Manuscript, 1992.

[AW08] S. Aaronson and A. Wigderson. Algebrization: A new barrier in complexity theory. In *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing*, pages 731–740, 2008.

[BFNW93] L. Babai, L. Fortnow, N. Nisan, and A. Wigderson, $BPP$ has subexponential time simulation unless $EXPTIME$ has publishable proofs, *Computational Complexity*, 3(4), 1993. pp. 193-219.

[BGS75] T. Baker, J. Gill, and R. Solovay. Relativizations of the P=?NP question. *SIAM Journal on Computing*, 4(4):431–442, 1975.

[B94] Paul Beame, A Switching Lemma Primer, manuscript, 1994.

[PBI93] Toniann Pitassi, Paul Beame, Russell Impagliazzo: Exponential Lower Bounds for the Pigeonhole Principle. *Computational Complexity*, Volume 3, 1993, pp.97-140.

[BF90] Donald Beaver, Joan Feigenbaum: Hiding Instances in Multioracle Queries. STACS 1990, pp. 37-48.

[BT03] A. Bogdanov and L. Trevisan, On worst-case to average-case reductions for $NP$ problems, FOCS, 2003, pp. 208-317.

[BT06] Andrej Bogdanov, Luca Trevisan: Average-Case Complexity. Foundations and Trends in Theoretical Computer Science (FTTCS) 2(1), 2006.

[DTT10] Anindya De, Luca Trevisan, Madhur Tulsiani: Time Space Tradeoffs for Attacks against One-Way Functions and PRGs. CRYPTO 2010: 649-665

[FF93] J. Feigenbaum and L. Fortnow, Random-self-reducibility of complete sets, *SIAM Jounral on Computing*, Vol. 22, 1993, pp. 994-1005.

[GT00] Gennaro, R., Trevisan, L.: Lower bounds on the efficiency of generic cryptographic constructions. In: Proceedings of the 41st IEEE Symposium on Foundations of Computer Science, pp. 305313 (2000)

[G93] Y. Gurevich, The Challenger-Solver Game: Variations on the Theme of $P = NP$. in: *Current Trends in Theoretical Computer Science – Essays and Tutorials*, Grzegorz Rozenberg and Arto Salomaa, ed., World Scientific Publishing Co., 1993, pp. 245-253.

[GST05] D. Gutfreund, R. Shaltiel, and A. Ta-Schma, If $NP$ languages are hard on the worst-case, then it is easy to find their hard instances. *Conference on Computational Complexity*, 2005.

[GKL93] Goldreich, O., Krawczyk, H. and Luby, M., "On the Existence of Pseudorandom Generators", *SIAM J. on Computing*, Vol. 22, No. 6, December, 1993, pp. 1163–1175.

[I95] Russell Impagliazzo: A Personal View of Average-Case Complexity. Structure in Complexity Theory Conference, 1995, pp. 134-147.

[I96] R. Impagliazzo, Very strong one-way functions and pseudo-random generators exist relative to a random oracle, manuscript, 1996.

[IKK] R. Impagliazzo, V. Kabanets, A. Kolokolova, An axiomatic approach toa algebraization, STOC 2009.

[IL90] R. Impagliazzo and L. Levin, No better ways to

generate hard $NP$ instances than picking uniformly at random, *FOCS*, 1990, pp. 538-545.

[IW97] R. Impagliazzo and A. Wigderson, $P = BPP$ if $E$ Requires Exponential Circuits: Derandomizing the XOR Lemma. STOC, 1997, pp. 220-229.

[Karp77] R. Karp, Probabilistic analysis of partitioning algorithms for the traveling salesman problem in the plane. *Mathematics of Operations Research*, Vol. 2, No. 3, 1977, pp. 209-224.

[KPW95] Jan Krajk, Pavel Pudl Alan R. Woods: An Exponenetial Lower Bound to the Size of Bounded Depth Frege Proofs of the Pigeonhole Principle. *Random Structures and Algorithms*, Vol:7, No. 1, 1995, pp. 15-40.)

[Levin86] L. Levin, Average case complete problems. *SIAM Jorunal on Computing*, 15(1), 1986, pp. 285-286.

[Lip91] R. Lipton, New Directions in Testing, *DIMACS Distributed Computing and Cryptography Workshop*, Vol. 2, p. 191, 1991.

[LFKN92] C. Lund, L. Fortnow, H. Karloff, and N. Nisan. Algebraic methods for interactive proof systems. *Journal of the Association for Computing Machinery*, 39(4):859–868, 1992.

[R93] A. Razborov, An equivalence between second order bounded domain bounded arithmetic and first order bounded arithmetic, In *Arithmetic, Proof Theory and Computational Complexity*, P.Clote and J. Krajicek, ed., Oxford University Press, 1993.

[RTV04] Omer Reingold, Luca Trevisan, Salil P. Vadhan: Notions of Reducibility between Cryptographic Primitives. TCC 2004:1-20

[TW87] Martin Tompa, Heather Woll, "Random self-reducibility and zero knowledge interactive proofs of possession of information," Foundations of Computer Science, Annual IEEE Symposium on, pp. 472-482, 28th Annual Symposium on Foundations of Computer Science (FOCS 1987), 1987.

[Wat10] Thomas Watson: Relativized Worlds without Worst-Case to Average-Case Reductions for NP. APPROX-RANDOM 2010, pp. 752-765.

[Yao90] Andrew Yao. Coherent functions and program checkers. In Proceedings of the 22nd ACM Symposium on Theory of Computing (STOC), 1990, pp. 84-94.

[Zim] M. Zimand, How to Privatize Random Bits, University of Rochester TR 616.

## APPENDIX

Here we give, for completeness, the proof of a generalized form of [I96], [Zim].

Let $1 \leq q(n), a(n) \leq 2^n$.

**Theorem 14.** *Let $D$ and $R$ be finite sets of size $N$. Let $A^f$ be a time $q$ oracle machine using $a$ bits of advice. With probability $1 - \delta$ over choice of a random 1-1 map $f$ from $D$ to $R$, for every $w$, there are at most $O(q(a+\log 1/\delta))$ strings $r$ in $R$ so that $f(A^f(r, w)) = r$.*

**Proof:** Our strategy is to bound the probability that there is a set of $t$ strings where the algorithm finds an inverse for each element of the set. The first attempt is to calculate the expected number of such sets. However, for most cases this expected number seems to be larger than 1. We then adopt a combinatorial trick of Goldreich, Kravjic and Luby [GKL93]: if the algorithm inverts $t >> s$ elements, then there must be *exponentially many* subsets of elements of size $s$ that it successfully inverts. We can then apply our bound on the number of such subsets and Markov's inequality.

Fix $A$ and advice string $w$. Let $s$ be an integer with $sq < N/2$. We also assume, without loss of generality, that $A^f$ queries its output before halting.

**Lemma 15.** *Assume $sq < N/2$. The expected number (over a random function $f$) of subsets $S$ of $R$ bit strings of size $s$ so that $f(A^f(y, w)) = y$ for each $y \in S$, is at most $(2e^2q)^s$.*

**Proof of Lemma:** Fix $S$. Simulate $A^f$ with advice $w$ on every element of $S$ (in some fixed order), and count how many distinct queries get mapped to some element of $S$. Then the events $E_i$ that the $i$'th distinct query gets answered by some element in $S$ have conditional probabilities at most $s/(N-iq) \leq s/(N-sq) \leq 2s/N$. There are at most $qs$ such queries total, and if $A^f$ inverts every element of $S$, there must be at least $s$ of these events that are true. Each of the $\binom{qs}{s}$ sets of $s$ of these events has probability at most $(2s/N)^s$ of all simultaneously occurring. So the probability that $A^f$ inverts every element of $S$ is at most $\binom{qs}{s}(2s/N)^s$ .

Summing over all sets $S$, the expected number of such sets is at most $\binom{qs}{s}(2s/N)^s\binom{N}{s} \leq (eqs/s)^s(2s/N)^s(eN/s)^s = (2e^2q)^s$. **QED (Lemma)**

Note that, if, for $s = (a + \log 1/\delta)$, $sq > N/2$, the theorem is trivially true, since the algorithm can be at most successful on all $N$ inputs. Now, if $sq < N/2$ and there is a set $S$ of size $t = 8e^2qs$ that $A^f$ inverts, then it also inverts any subset of $S$, and in particular, there are at least $\binom{t}{s} \geq (t/s)^s = (8e^2q)^s$ such subsets of size $s$. By the above Lemma and Markov's inequality, this happens with probability at most $(2e^2q)^s/(8e^2q)^s = 4^{-s}$. In particular, for $s = a + \log 1/\delta$, we can sum up over all $2^{a(n)}$ advice strings, to get a total probability of at most $\delta$ that there is an advice string of length $a$ for which $A^f$ finds pre-images for $8e^2q(a + \log 1/\delta)$ strings. **QED**