Some Results on Average-Case Hardness Within the Polynomial Hierarchy

Some Results on Average-Case Hardness within the Polynomial Hierarchy

A. Pavan[1⋆], Rahul Santhanam[2], and N. V. Vinodchandran[3⋆⋆]

[1] Department of Computer Science, Iowa State University
[2] Department of Computer Science, Simon Fraser University
[3] Department of Computer Science and Engineering, University of Nebraska-Lincon

Abstract. We prove several results about the average-case complexity of problems in the Polynomial Hierarchy (PH). We give a connection among average-case, worst-case, and non-uniform complexity of optimization problems. Specifically, we show that if P^{NP} is hard in the worst-case then it is either hard on the average (in the sense of Levin) or it is non-uniformly hard (i.e. it does not have small circuits).
Recently, Gutfreund, Shaltiel and Ta-Shma (*IEEE Conference on Computational Complexity, 2005*) showed an interesting worst-case to average-case connection for languages in NP, under a notion of average-case hardness defined using uniform adversaries. We show that extending their connection to hardness against quasi-polynomial time would imply that NEXP doesn't have polynomial-size circuits.
Finally we prove an unconditional average-case hardness result. We show that for each k, there is an explicit language in P^{Σ_2} which is hard on average for circuits of size n^k.

1 Introduction

Average-case complexity is one of the central concepts in complexity theory. There are several different reasons for studying average-case complexity. The notion of being hard on average is fundamental to cryptography [Gol01,Gol04], since the security of most cryptographic protocols is conditioned on the assumption that certain problems such as factoring and discrete logarithm problem are hard on average. Notions of average-case complexity also appear naturally in the theory of pseudo-randomness [BM84,Yao82], learning theory [JS05] and the study of heuristics for solving NP-complete problems [ART06].

Does the existence of a worst-case hard problem (say, with respect to polynomial-size circuits) in a complexity class $\mathcal{C}$ imply the existence of an average-case hard problem in the class? This question is particularly significant for the case of NP in part because of its connections to cryptography and the theory of approximation. Answering the question positively for NP would enable us to base cryptog-

⋆ Research supported in part by NSF grant CCF-0430807, and a Big12 Faculty Fellowship.
⋆⋆ Research supported inpart by NSF grant CCF-0430991.

raphy on NP-hardness rather than on the hardness of specific algebraic/number-theoretic problems such as discrete logarithm and factoring. From the perspective of hardness of approximation, there is a recent line of work [Fei02,Ale03] showing that average-case hardness of NP problems would imply much better inapproximability results for certain natural problems in NP than that are currently known.

For "large enough" complexity classes such as EXP and PSPACE, it is known that worst-case hardness implies average-case hardness. This follows from generic hardness amplification techniques [STV01,TV02] which were developed in the context of the theory of pseudo-randomness. However, for NP or other classes in the polynomial-time hierarchy (PH), the techniques that work for EXP or PSPACE are known to fail [BT03,Vio05], and the question of whether worst-case hardness implies average-case hardness for NP and PH remains unsolved.

Our Results

We consider various notions of average-case hardness that have been defined in the literature, and investigate the possibility of constructing languages within the polynomial-time hierarchy that are average-case hard according to these notions.

First, we consider Levin's framework for average-case complexity [Lev86]. Informally, in this framework, a problem L is easy on average if for every polynomial-time samplable distribution μ, there is some algorithm which solves L and halts in polynomial time with high probability over the distribution μ. It is a longstanding open problem whether the assumption that NP is easy on average under this notion implies NP = P. It is also not known if there is an oracle under which the implication would not follow (and hence would require a non-relativizing technique to prove, assuming it is true). The analogous question for Σ_k^{P}, for any $k > 1$ also remains open.

We ask a weaker question: is there any non-trivial easiness assumption about PH which in conjunction with the assumption about easiness on average imply that NP = P? A natural assumption to consider is the assumption of *non-uniform* easiness, i.e., solvability by polynomial-size circuits. Our first theorem is along this direction.

Theorem 1.1. *If* $\mathrm{NP} \neq \mathrm{P}$*, then* $\mathrm{NP} \not\subseteq \mathrm{P/poly}$ *or* P^{NP} *is average-case hard.*

An immediate corollary of this result is that if $\mathrm{P}^{\mathrm{NP}} \neq \mathrm{P}$, then either P^{NP} is non-uniformly hard or P^{NP} is average-case hard. P^{NP} has a natural interpretation as the class of optimization problems whose decision versions are in NP, thus we get that for optimization problems, worst-case hardness implies either average-case hardness or non-uniform hardness.

The nonuniform hardness in the above theorem refers to worst-case nonuniform hardness. We consider the possibility of improving this to average-case nonuniform hardness. As our second main result, we obtain the following improvement to the above theorem (for a more precise statement see Section 3).

Theorem 1.2. *If* NP $\neq$ P*, then either* P^{NP} *is average-case hard, or there is a language* L *in* NP *such that for every* k *there is a polynomial-time samplable distribution* μ *and* L *is average-case hard for* n^k*-size circuits with respect to distribution* μ*.*

A more restrictive notion (than the one due to Levin) of average-case easiness that has gained prominence recently is the notion of easiness against uniform adversaries. A language L is said to be easy against a class $\mathcal{C}$ of adversaries if there is a fixed polynomial-time simulation of L such that no adversary in $\mathcal{C}$ outputs an instance on which the simulation differs from L. Note that in this setting, the simulation is independent of the adversary. This is more restrictive than Levin's notion, since in Levin's setting the running time of the simulation can depend on the running time of the adversary. A recent work of Gutfreund, Shaltiel and Ta-Shma [GSTS05] shows that under this notion the average-case complexity of NP is same as its worst-case complexity. In particular they show that if every language in NP is easy against polynomial-time adversaries then in fact NP = P. Their result has been further refined by Atserias [Ats06].

The question we ask is: how relevant is the technique of [GSTS05] for proving an average-case to worst-case connection for NP in Levin's framework? In particular if we allow the simulation to run for more than polynomial time (say quasi-polynomial time) can we extend the results of Gutfreund, Shaltiel and Ta-Shma to get an average-case to worst-case equivalence for NP? We show that such a result would imply a groundbreaking circuit lower bound result, and hence is unlikely to be provable using current techniques.

Theorem 1.3. *If* NP $\subseteq$ $quasi_{\mathrm{P}}-\mathsf{QP}$ *implies* NP $\subseteq \mathsf{QP}$*, then* NEXP $\not\subseteq$ P/poly.

We refer the reader to Section 2 for the definition of "$quasi_{\mathrm{P}}$", which formalizes the notion of easiness used in [GSTS05].

Next we consider the question of whether known worst-case lower bounds in PH can be extended to average-case lower bounds. To be specific, we ask: which level in the polynomial-time hierarchy has languages that are average-case hard for circuits of size n^k? Kannan [Kan82] showed using a nonconstructive argument that for any fixed k, there is a language in the second level of PH (more precisely $\Sigma_2^{\mathrm{P}} \cap \Pi_2^{\mathrm{P}}$) that cannot not be computed by circuits of size n^k. Since then there have been a series of attempts to prove better upper bounds on the complexity of such a language. The current best upper bound known [Cai01] is $\mathrm{S}_2^{\mathrm{P}}$, which is a subclass of $\Sigma_2^{\mathrm{P}} \cap \Pi_2^{\mathrm{P}}$. There has been related work on constructing explicit languages in low levels of PH that do not have circuits of size n^k (the $\mathrm{S}_2^{\mathrm{P}}$ upper bound is proved non-constructively). Miltersen, Vinodchandran and Watanabe [MVW99] showed a constructive upper bound of $\mathrm{P}^{\Sigma_2^{\mathrm{P}}}$; Cai and Watanabe [CW03] improved the upper bound to Σ_2.

Note that since an average-case to worst-case connection is not known within PH, we cannot directly use the results above to show an average-case hardness result. Nevertheless, we strengthen the technique of Miltersen, Vinodchandran and Watanabe to show that for each k, there is an explicit language in $\mathrm{P}^{\Sigma_2^{\mathrm{P}}}$

which cannot be *approximated* on significantly more than 1/2 the inputs of any input length by circuits of size n^k (refer to Section 2 for the precise definition of (n^k, n^k)-hard in the following).

Theorem 1.4. *For each k, there is a language L_k in $\mathrm{P}^{\Sigma_2^{\mathrm{P}}}$ such that L_k is (n^k, n^k)-hard.*

2 Preliminaries

We say that $\mu = (\mu_1, \mu_2, \cdots)$ is an *ensemble of distributions* if each μ_n is a distribution over Σ^n. We often use the word distribution instead of ensemble of distributions. A distribution μ is p-samplable if there is a probabilistic algorithm A such that for every $x \in \Sigma^n$

$$Pr[A(1^n) = x] = \mu_n(x).$$

We use Levin's notion of *average polynomial-time* [Lev86]. In his definition of average-polynomial time, Levin considered a distribution over Σ^* rather than an ensemble of distributions. However, many times it is more convenient to consider an ensemble of distributions rather than a single distribution. Gurevich [Gur91] and Impagliazzo [Imp95] showed that Levin's definition can be adapted to the case of an ensemble of distributions. We follow this adaptation.

Let $\mu = (\mu_1, \mu_2, \cdots)$ be an ensemble of distributions. We associate distribution μ_{assoc} over Σ^*, to the ensemble $(\mu_1, \mu_2, \cdots)$, as follows: if x is a string of length n, then

$$\mu_{assoc}(x) = \frac{6}{\pi^2}\frac{1}{n^2}\mu_n(x).$$

Definition 2.1. *([Lev86]) Let L be a language and $\mu = (\mu_1, \mu_2, \cdots)$ be a distribution. We say (L, μ) is in* Average-P *if there is a machine M that decides L and a constant $k \geqslant 1$,*

$$\sum_x \frac{(T_M(x))^{1/k}}{|x|}\mu_{assoc}(x) < \infty.$$

Remark. In Levin's notion of Average polynomial-time, a single distribution over Σ^* is used instead of an ensemble of distributions as above.

We find the following observation to be useful.

Observation 2.2 *Let μ be an ensemble of distributions, and let (L, μ) in* Average-P*. There exists a Turing machine M that accepts L such that for every polynomial $p(.)$, there exists a constant $l > 0$, and for all but finitely many n,*

$$\Pr_{x \in \mu_n}[M(x) \text{ does not halt in } n^l \text{ steps}] < 1/p(n).$$

Given a complexity class $\mathcal{C}$, let Dist$\mathcal{C}$ denote the class of distributional problems (L, μ), where $L \in \mathcal{C}$ and μ is a p-samplable ensemble. Now whether Dist$\mathcal{C} \subseteq$ Average-P is the average-case analogue of whether $\mathcal{C} \subseteq$ P. Given a class $\mathcal{C}$, we say that $\mathcal{C}$ *is easy on average* if Dist$\mathcal{C} \subseteq$ Average-P.

We can adapt Levin's notion of average polynomial time to function classes also. The following observation is easy to prove.

Observation 2.3 *If* P^{NP} *is easy on average, then* $\mathsf{PF}^{\mathrm{NP}}$ *is easy on average.*

We also consider the notions of average-case complexity under the uniform distribution in nonuniform models of computation.

Definition 2.4. *Let s and h be functions from $\mathbb{N}$ to $\mathbb{N}$. A language L is called (s, h)-hard if for every $s(n)$-size circuit family $C = (C_1, C_2, \cdots)$*

$$\Pr_{x \in \Sigma^n}[L(x) = C_n(x)] \leqslant 1/2 + 1/h(n).$$

Here x is drawn uniformly at random from Σ^n.

Definition 2.5. *A distributional problem (L, μ) is in $\mathsf{HSIZE}(n^k)$, if for every polynomial p, there is a n^k-size circuit family $C = (C_1, C_2, \cdots)$ such that for all but finitely many n*

$$\Pr_{x \in \mu_n}[L(x) \neq C_n(x)] \leqslant 1/p(n).$$

In this paper, we also study a notion of easy on average that is different from Levin's notion of easy on average. This notion naturally arises in the theory of pseudo-randomness and uniform derandomization. This notion was implicit in the work of Impagliazzo and Wigderson [IW98]. Kabanets [Kab01] made this explicit and defined "pseudo classes."

Definition 2.6. *([Kab01]) Let $\mathcal{C}$ be a complexity class. A language L is in pseudo$_{\mathrm{P}}-\mathcal{C}$ if there is a language L' in $\mathcal{C}$ such that for every polynomial-time machine R for all but finitely many n, $R(1^n) \notin L \Delta L'$.*

Thus if a language L is in pseudo$_{\mathrm{P}}-\mathcal{C}$, there is a simulation L' for L and no adversary R can find places where L' and L differ. We obtain the class quasi$_{\mathrm{P}}-$P (defined in [vMS05]) by allowing the adversary R to output more than one string.

Definition 2.7. *A language L is in quasi$_{\mathrm{P}}-\mathcal{C}$, if there is a language L' in $\mathcal{C}$ such that for every polynomial-time machine R for all but finitely many n, no output of $R(1^n)$ belongs to $L \Delta L'$.*

A consistent circuit for SAT is a circuit that can err only on one-side. More formally,

Definition 2.8. *We say a circuit C is* consistent circuit *for* SAT*, if C outputs a satisfying assignment whenever it says a formula is satisfiable.*

We use the following known results in our proofs.

Theorem 2.9. *[BCK+96] Assume* NP $\subseteq$ P/poly. *There is a* $\mathrm{ZPP}^{\mathrm{NP}}$ *machine* M *such that* M *on input* 1^n *either outputs "?" or outputs a circuit for* SAT_n. *Probability that* M *outputs "?" is at most* $1/2^n$.

Theorem 2.10. *[FPS03] For every* k, *there is a* $\mathrm{ZPP}^{\mathrm{NP}}$ *algorithm* M *such that if* SAT *does not have* n^{k+2}*-size circuits at length* n *then* M *on input* 1^n *either outputs "?" or outputs a list of formulas* $\phi_1, \phi_2, \cdots, \phi_m$, $m \leqslant n^{2k}$, *such that*

- $\Pr[M(1^n) =?] \leqslant 1/2^n$
- *If* $M(1^n)$ *outputs* $\phi_1, \cdots \phi_m$, *then for every* n^k*-size consistent circuit* C, *there exists* i, $1 \leqslant i \leqslant m$ *such that* $C(\phi_i) \neq \mathrm{SAT}(\phi_i)$.

3 Easiness on Average versus Nonuniform Easiness

In this section we show results that connect the worst-case, average-case, and non-uniform hardness of the class P^{NP}.

Theorem 1.1 *If* P $\neq$ NP, *then at least one of the following statements is true.*

- P^{NP} *is not easy on average.*
- NP *is not in* P/poly.

Proof. Assume that NP is in P/poly and P^{NP} is easy on average. Since NP is in P/poly, by Theorem 2.9 there is a $\mathrm{ZPP}^{\mathrm{NP}}$ machine M that on input 1^n outputs a circuit for SAT_n with high probability. Assume that $M(1^n)$ needs n^k random bits. Define a function f as follows:

$$f(1^n, r) = M(1^n, r).$$

where $|r| = n^k$, and $M(1^n, r)$ is the output of M when it is given r as random seed. Clearly, $f \in \mathsf{PF}^{\mathrm{NP}}$. Since P^{NP} is easy on average, by Observation 2.3, for every p-samplable distribution μ, (f, μ) can be computed in polynomial-time on average. Consider the following distribution $\mu = (\mu_1, \mu_n, \cdots)$, where μ_n randomly and uniformly picks a string r of length n^k and outputs $\langle 1^n, r\rangle$.

Let N be a machine that computes f in average polynomial time with respect to μ. By Observation 2.2, there exists $l > 0$ such that

$$Pr_r[N(1^n, r) \text{ does not halt in } n^l \text{ steps}] \leqslant 1/n^2.$$

Since N computes f

$$Pr_r[N(1^n, r) =?] \leqslant 1/2^n.$$

Thus if we randomly pick r, probability that $N(1^n, r)$ either takes more than n^l time or outputs "?" is at most $1/n$. Thus if we randomly pick r and stop the computation of $N(1^n, r)$ after n^l steps, then with very high probability it outputs a circuit for SAT_n. This gives a probabilistic polynomial-time algorithm that can compute circuits for SAT.

Thus SAT $\in$ BPP and so NP $\subseteq$ BPP. Buhrman, Fortnow and Pavan [BFP05] showed that if NP is easy on average, then BPP = P. Thus NP = P.

This theorem has the following interesting corollary.

Corollary 3.1. *If* P^{NP} *is hard in the worst-case, then either it is non-uniform hard or average-case hard.*

Theorem 1.1 says that if NP does not equal to P, then either P^{NP} is hard on average or there is a language in NP that is not in $\mathsf{SIZE}(n^k)$ for every $k > 1$. This language in NP is worst-case hard in the non-uniform model. Can we make this language to be average-case hard in the non-uniform model? We show the following:

Theorem 1.2 *If* $\mathrm{P} \neq \mathrm{NP}$*, then at least one of the following statements is true.*

- P^{NP} *is not easy on average.*
- *There is a language* L *in* NP *such that for every* k *there is a p-samplable distribution* μ *such that* $(L, \mu) \notin \mathsf{HSIZE}(n^k)$.

Remark. In this result, the distribution μ depends on the constant k. Making the distribution independent of k yields the much sought average-case to worst-case connection for P^{NP}.

The theorem follows from the following two Lemmas. We omit the proofs due to lack of space. Proof of these Lemma 3.2 makes crucial use of Theorem 2.10.

Lemma 3.2. *If* $\mathrm{P} \neq \mathrm{NP}$*, then at least one of the following statements is true.*

- P^{NP} *is not easy on average.*
- *For every* k *there is a p-samplable distribution* μ*, and infinitely many* n *such that for every* n^k*-size consistent circuit fancily* $C = (C_1, C_2, \cdots)$ *for* SAT

$$\Pr_{x \in \mu_n}[C_n(x) \neq \mathrm{SAT}(x)] \geqslant 1/n^{4k}.$$

Lemma 3.3. *Assume that the following statement holds: For every* k *there is a p-samplable distribution* μ*, and infinitely many* n *such that for every* n^k*-size consistent circuit family* $C = (C_1, C_2, \cdots)$ *for* SAT*,*

$$\Pr_{x \in \mu_n}[C_n(x) \neq \mathrm{SAT}(x)] \geqslant 1/n^{4k}.$$

Then, there is a language L *in* NP *such that for every* k*, there is a p-samplable distribution* μ *and*

$$(L, \mu) \notin \mathsf{HSIZE}(n^k).$$

4 On the Difficulty of Showing Easiness on Average Implies Easiness in the Worst Case

A recent result by Gutfreund, Shaltiel and Ta-Shma [GSTS05] on worst-case to average-case reductions for NP problems states that if there is a simulation of SAT in polynomial time which fools all polynomial-time adversaries, then NP = P.

Theorem 4.1. *If* NP $\subseteq$ $quasi_{\mathrm{P}}$−P*, then* NP = P*.*

Theorem 4.1 can be interpreted as follows. If SAT is not in polynomial time, then for any polynomial-time algorithm A purporting to solve SAT, there is an adversary—a polynomial time procedure—that for each n produces a small list of candidate *counter-examples* of size n. Namely the adversary outputs a list of formulae such that there is at least one formula ϕ in the list for which $A(\phi) \neq SAT(\phi)$. In fact, the proof of Theorem 4.1 gives an upper bound of 3 on the size of the list.

It is crucial to the proof of Theorem 4.1 that the adversary has more resources than the simulating class. Indeed, the proof of Theorem 4.1 proceeds via construction of an adversary which simulates an algorithm A purporting to solve SAT. On the other hand, showing an average-case to worst-case connection for NP under Levin's notion would mean that if NP $\neq$ P, then there is a distribution μ such that (SAT, μ) is not solved on average by any polynomial-time algorithm, where the algorithm may take more time than is required to sample from μ. Thus intuitively, if the method of [GSTS05] is to be applicable to showing an average-case to worst-case connection for NP, it should be possible to extend Theorem 4.1 to a setting where the simulating class has more power than the adversary. We show that this is unlikely using current techniques (indeed, using relativizing techniques) since NEXP $\not\subseteq$ P/poly is a consequence.

We will actually show that NEXP $\neq$ MA, which implies the circuit lower bound by the following result of Impagliazzo, Kabanets and Wigderson:

Theorem 4.2. *[IKW02]* NEXP $\neq$ MA *if and only if* NEXP $\not\subseteq$ P/poly*.*

We consider two cases, the first where NP is somewhat easy in the worst case, and the second where NP is somewhat hard according to the notion of hardness in [GSTS05]. In both cases, we show that MA $\neq$ NEXP follows. Thus MA $\neq$ NEXP would follow from an average-case to worst-case connection. In the first case, we use standard techniques, and in the second case, we use the "easy witness" method of Kabanets [Kab01] and Impagliazzo, Kabanets and Wigderson [IKW02]. Let QP denote the class of languages that can be decided in deterministic quasi-Polynomial time, and NQP is the nondeterministic analogue of QP.

Lemma 4.3. *If* NP $\subseteq$ QP*, then* MA $\neq$ NEXP*.*

Proof. We will prove something even stronger, namely that MA $\neq$ EXP.

By Lautemann's theorem [Lau83], MA $\subseteq \Sigma_2^{\mathrm{P}}$. If NP $\subseteq$ QP, then MA $\subseteq \Sigma_2^{\mathrm{P}} = \mathrm{NP}^{\mathrm{NP}} \subseteq \mathrm{NP}^{\mathsf{QP}} \subseteq$ NQP.

By padding, if NP $\subseteq$ QP, then NQP = QP, and hence MA $\subseteq$ QP. By the hierarchy theorem for deterministic time, EXP $\not\subseteq$ QP, and hence MA $\neq$ EXP.

Next, we show that a superpolynomial lower bound on average-case hardness in the framework of [GSTS05] would also separate MA and NEXP. We will need the optimal construction of pseudo-random generators due to Umans [SU05,Uma02] in the proof.

Theorem 4.4. *There is a function* $G : \{0,1\}^{2^m} \times \{0,1\}^{O(m)} \to \{0,1\}^{m^s}$ *computable in polynomial time such that if* f *is a Boolean function on* m *bits which doesn't have circuits of size* m^{3s}*, then for any circuit* C *of size* m^s*,* $|\Pr_{y\in\{0,1\}^{m^s}}(C(y) = 1) - \Pr_{x\in\{0,1\}^m}(C(G(f, x)) = 1)| < 1/m^s$

Lemma 4.5. *If* NP $\not\subseteq$ *quasi*$_{\mathrm{P}}-\mathsf{QP}$*, then* MA $\neq$ NEXP*.*

Proof Sketch. Fix a language L in NP. We attempt to simulate L in deterministic time $2^{polylog(n)}$ on inputs of length n as follows. For an input x of length n, we interpret a witness for x as the truth table of a Boolean function (rounding the witness size upwards to a power of 2). We search for witnesses describable by small circuits, i.e., circuits of size $polylog(n)$. If we find such a witness for x, we accept x, otherwise we reject. Clearly, the search can be implemented exhaustively in time $2^{polylog(n)}$.

Since NP $\not\subseteq$ *quasi*$_{\mathrm{P}} - \mathsf{QP}$, there is an $L \in$ NP such that the simulation above fails for L. Moreover, there is a polynomial time machine B outputting a list of instances such that the simulation fails on at least one of the instances. We will use the machine B to derive a simulation of MA in non-deterministic subexponential time with small advice, and then use a hierarchy theorem to show that this implies a separation of MA and NEXP.

We show that for any language $L' \in$ MA, $L' \in i.o.\mathrm{NTIME}(2^{O(m)})/O(m)$. The basic idea is that the machine B can be used to derandomize a Merlin-Arthur machine accepting L' infinitely often, given small advice. This is because for infinitely many input lengths n, there is at least one instance $y \in L$ of length n output by B such that none of the witnesses for y are describable by small circuits. Thus, if we knew y, we could non-deterministically compute the truth table of a hard function by merely guessing and verifying a witness for y. Once we have the truth table of a hard function, we could use Theorem 4.4 to derandomize a polynomial-time Merlin-Arthur machine and simulate it's computation in NTIME$(2^{O(m)})$, where m is the length of the input to the machine.

We do not know y but B does produce a small list containing y. Thus, given a small amount of advice telling us the index of y in the list, we can determine y. We also do not know precisely for which input lengths B produces a list containing at least instance in L with hard witnesses. But we know that this happens infinitely often, and we can again use a small amount of advice to point to the right input lengths. We omit the details in this sketch.

Now assume, for the purpose of contradiction, that MA = NEXP. Since MA $\subseteq$ EXP $\subseteq$ NEXP, we have that EXP = NEXP. This implies that there is some constant c such that NE $\subseteq$ DTIME(2^{n^c}) (since NE has a complete language, and a deterministic time upper bound for that complete language also holds for any language in NE). It follows that NE$/O(n) \subseteq$ DTIME$(2^{n^c})/O(n)$. We have that MA $\subseteq i.o.$NE$/O(n) \subseteq i.o.$DTIME$(2^{n^c})/O(n)$. Since MA = EXP by assumption, we have that EXP $\subseteq i.o.$DTIME$(2^{n^c})/O(n)$, which is a contradiction to the time hierarchy theorem for deterministic time. □

Now, Theorem 1.3 follows from above two lemmas.

Theorem 1.3 *. If* NP $\subseteq$ *quasi*$_{\mathrm{P}}-\mathsf{QP}$ *implies* NP $\subseteq \mathsf{QP}$*, then* NEXP $\not\subseteq$ P/poly*.*

Proof. By assumption, either NP $\subseteq$ QP or NP $\not\subseteq$ $quasi_{\mathrm{P}}-\mathsf{QP}$. In the first case, by Lemma 4.3, MA $\neq$ NEXP. In the second case also, by Lemma 4.5, MA $\neq$ NEXP. Thus, in either case, MA $\neq$ NEXP, which implies NEXP $\not\subseteq$ P/poly by Theorem 4.2.

5 Average-case circuit lower bounds within PH

Kannan [Kan82] showed that for every k, there exist functions in the polynomial-time hierarchy for which no n^k-size circuits exist. However, this is a worst-case hardness result. Are there functions in PH that are hard on average for n^k-size circuits?

We show how to find such functions in the third level of the PH.

Theorem 5.1. *For any k and h, there is a language $L \in \mathrm{P}^{\Sigma_2^P}$ that is (n^k, n^h) hard.*

We first start with a function $g : \{0,1\}^{2(k+h)\log n} \to \{0,1\}$ that is (n^k, n^h) hard and then randomly pad the input to get a function f on n bits with the same hardness.

Theorem 5.2. *There is a function $g : \{0,1\}^{2(k+h)\log n} \to \{0,1\}$ that is (n^k, n^h) hard. Moreover, there is an $\mathrm{FP}^{\Sigma_2^P}$ procedure that outputs the lexicographically first such function.*

Proof. Consider a random function from $\{0,1\}^{2(k+h)\log n} \to \{0,1\}$ viewed as a Boolean string of length $n^{2(k+h)}$. Fix a circuit C of size n^k. The expected agreement between C and g is $\frac{n^{2(k+h)}}{2}$. Thus using Chernoff's bounds, $\Pr((C(x) = g(x)) > (1+\delta)\frac{n^{2(k+h)}}{2}) \leqslant e^{\frac{-\delta^2 n^{2(k+h)}}{6}}$. For $\delta = \frac{1}{n^h}$, this probability is $< 2^{-n^{2k}}$. There are at most $2^{n^{k+1}}$ circuits of size $\leqslant n^k$. Thus by union bound there exists a function $g : \{0,1\}^{2(k+h)\log n} \to \{0,1\}$ that is (n^k, n^h) hard.

Since the function is on $O(\log n)$ size inputs, it is easy to see that an $\mathrm{FP}^{\Sigma_2^P}$ procedure can output the lexicographically first such function.

Proof. (*Of Theorem 5.1*). Consider the function $f : \{0,1\}^n \to \{0,1\}$ defined as follows. Let $x = yz$ where y is the first $2(k+h)\log n$ bits of x. Define $f(x) = g(y)$ where g is the hard function from the above theorem. Claim is that the function f is (n^k, n^s) hard. For a contradiction, let D be a circuit of size at most n^k so that $\Pr_x((D(x) = f(x)) > \frac{1}{2} + \frac{1}{n^h})$. That is, $\Pr_{yz}((D(yz) = f(yz)) > \frac{1}{2} + \frac{1}{n^h})$. Then by an averaging argument there is a z so that $\Pr_y((D(yz) = f(yz)) > \frac{1}{2} + \frac{1}{n^h})$. Thus by hardwiring this z into D , we get a circuit D_z of size $\leqslant n^k$ so that $\Pr_y((D_z(y) = g(y)) > \frac{1}{2} + \frac{1}{n^h})$. This contradicts the hardness of g.

Theorem 1.4 is a special case of Theorem 5.1.

Acknowledgements

We thank the friendly staff of Iowa Western Community College, Atlantic, IA for providing facilities where a part of this work was done.

References

[Ale03] M. Alekhnovich. More on average case vs approximation complexity. In *Proceedings of 44th IEEE Symposium on Foundations of Computer Science*, pages 298–307, 2003.

[ART06] D. Achlioptas and F. Ricci-Tersenghi. On the solution-space geometry of random constraint satisfaction problems. In *Proceedings of Symposium on Theory of Computing*, page to appear, 2006.

[Ats06] A. Atserias. Non-uniform hardness for NP via black-box adversaries. In *Proceedings of Conference on Computational Complexity*, page to appear, 2006.

[BCK+96] N. Bshouty, R. Cleve, S. Kannan, R. Gavalda, and C. Tamon. Oracles and queries that are sufficient for exact learning. *Journal of Computer and System Sciences*, 52:421–433, 1996.

[BFP05] H. Buhrman, L. Fortnow, and A. Pavan. Some results on derandomization. *Theory of Computing Systems*, 38(2):211–227, 2005.

[BM84] M. Blum and S. Micali. How to generate cryptographically strong sequences of pseudo-random bits. *SIAM J. Comput.*, 13:850–864, 1984.

[BT03] A. Bogdanov and L. Trevisan. On worst-case to average-case reductions for np problems. In *Proceedings of the 44th IEEE Conference on Foundations of Computer Science*, pages 308–317, 2003.

[Cai01] J. Cai. $S_2^p \subseteq \mathrm{ZPP}^{\mathrm{NP}}$. In *Proceedings of the 42nd Annual Symposium on Foundations of Computer Science, 2001*, pages 620–629, 2001.

[CW03] J. Cai and O. Watanabe. On proving circuit lower bounds against the polynomial hierarchy: Positive and negative results. In *Proceedings of Ninth Annual International Conference on Combinatorics and Computing*, pages 202–211, 2003.

[Fei02] Uriel Feige. Relations between average case complexity and approximation complexity. In *Proceedings of 35th Annual ACM Symposium on Theory of Computing*, pages 534–543, 2002.

[FPS03] L. Fortnow, A. Pavan, and S. Sengupta. Proving SAT does not have small circuits with an application to the two queries problem. In *Proceedings of the 18th IEEE Conference on Computational Complexity*, pages 347–350, 2003.

[Gol01] O. Goldreich. *Foundations of Cryptography - Volume 1*. Cambridge University Press, 2001.

[Gol04] O. Goldreich. *Foundations of Cryptography - Volume 2*. Cambridge University Press, 2004.

[GSTS05] D. Gutfreund, R. Shaltiel, and A. Ta-Shma. If NP languages are hard on the worst-case then it is easy to find their hard instances. In *IEEE Conference on Computational Complexity*, pages 243–257, 2005.

[Gur91] Y. Gurevich. Average case completeness. *Journal of Computer and System Sciences*, 42:346–398, 1991.

[IKW02] R. Impagliazzo, V. Kabanets, and A. Wigderson. In search of an easy witness: Exponential time vs Probabilistic polynomial time. *Journal of Computer and System Sciences*, 65:672–694, 2002.

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

[IW98] R. Impagliazzo and A. Wigderson. Randomness vs. time: de-randomization under a uniform assumption. In *39th Annual Symposium on Foundations of Computer Science: proceedings, 1998*, pages 734–743, 1998.

[JS05] J. Jackson and R. Servedio. On learning random DNF formulas under the uniform distribution. In *Proceedings of 9th International Workshop on Randomness and Computation*, pages 342–353, 2005.

[Kab01] V. Kabanets. Easiness assumptions and hardness tests: trading time for zero error. *Journal of Computer and System Sciences*, 63(2):236–252, 2001.

[Kan82] R. Kannan. Circuit-size lower bounds and non-reducibility to sparse sets. *Information and Control*, 55:40–56, 1982.

[Lau83] C. Lautemann. BPP and the polynomial hierarchy. *Information Processing Letters*, 17:215–217, November 1983.

[Lev86] L. Levin. Average case complete problems. *SIAM Journal of Computing*, 15:285–286, 1986.

[MVW99] P. B. Miltersen, N. V. Vinodchandran, and O. Watanabe. Super-polyomial versus half-exponential circuit size in the exponential hierarchy. In *Proceedings of Fifth Annual International Conference on Computing and Combinatorics*, pages 210–220, 1999.

[STV01] M. Sudan, L. Trevisan, and S. Vadhan. Pseudorandom generators without the XOR lemma. *JCSS: Journal of Computer and System Sciences*, 62, 2001.

[SU05] R. Shaltiel and C. Umans. Simple extractors for all min-entropies and a new pseudorandom generator. *Journal of the ACM*, 52, 2005.

[TV02] L. Trevisan and S. Vadhan. Pseudorandomness and average-case complexity via uniform reductions. In *Annual IEEE Conference on Computational Complexity*, volume 17, 2002.

[Uma02] C. Umans. Pseudo-random generators for all hardnesses. In *Symposium on Theory of Computing*, pages 627–634, 2002.

[Vio05] E. Viola. On constructing parallel pseudorandom generators from one-way functions. In *Proceedings of the 20th IEEE Conference on Computational Complexity*, 2005.

[vMS05] D. van Melkebeek and R. Santhanam. Holographic proofs and derandomization. *SIAM Journal on Computing*, 35(1):59–90, 2005.

[Yao82] A. Yao. Theory and applications of trapdoor functions. In *Proc. 23rd IEEE Symposium on Foundations of Computer Science*, pages 80–91, 1982.