NP with Small Advice

Lance Fortnow
Department of Computer Science
University of Chicago
Chicago, IL 60637
fortnow@cs.uchicago.edu

Adam R. Klivans
TTI-Chicago
Chicago, IL 60637
klivans@tti-c.org

February 20, 2005

Abstract

We prove a new equivalence between the non-uniform and uniform complexity of exponential time. We show that $\mathsf{EXP} \subseteq \mathsf{NP}/\log$ if and only if $\mathsf{EXP} = \mathsf{P}^{\mathsf{NP}}_{||}$. Our equivalence makes use of a recent result due to Shaltiel and Umans showing EXP in $\mathsf{P}^{\mathsf{NP}}_{||}$ implies EXP in $\mathsf{NP}/\mathsf{poly}$.

1 Introduction

Let A and B be uniform complexity classes such that $\mathsf{B} \subseteq \mathsf{A}$. If A seems much "larger" than B then it is often the case that we can prove that B is *strictly* contained in A, e.g. let $\mathsf{B} = \mathsf{P}$ and $\mathsf{A} = \mathsf{EXP}$. Is the same true if we consider a *non-uniform* analogue of B? That is to say, augment B by giving it access to some advice string b such that b depends only on the length of x; can we still separate A from B/b? If not, can we derive interesting consequences on A if it is contained in B/b, i.e. can we show that A collapses to some smaller complexity class?

These questions are of central importance in computational complexity theory, particularly in the area of derandomization, where both separations of uniform from non-uniform classes or collapses of uniform classes have important consequences:

Separations

- If $\mathsf{EXP} \not\subseteq \mathsf{P/poly}$, then Babai et al. [2], building on the "Hardness versus Randomness" paradigm [20], have shown that BPP is contained in subexponential time and that MA is contained in non-deterministic subexponential time (both containments are for infinitely many input lengths).
- It is known that if EXP cannot be computed by nondeterministic polynomial-size circuits then it is possible to obtain similar derandomizations of AM [16, 19, 22]. Shaltiel and Umans [21] were the first to prove that if $\mathsf{EXP} \not\subseteq \mathsf{NP/poly}$ then $\mathsf{AM} \subseteq \mathsf{NSUBEXP}$ for infinitely many input lengths.

Collapses

Perhaps less well known than the above derandomizations are equally important results showing that uniform complexity classes such as EXP or NEXP collapse if they are contained in smaller, non-uniform classes:

- Babal et al. [2] showed that $\mathsf{EXP} \subseteq \mathsf{P/poly}$ implies that $\mathsf{EXP} = \mathsf{MA}$, improving on work due to Meyer [15] who first proved that $\mathsf{EXP} \subseteq \mathsf{P/poly}$ implies $\mathsf{EXP} = \Sigma_2^{\mathsf{P}}$.
- Impagliazzo et al. [12] improved the above collapse and showed that $\mathsf{NEXP} \subseteq \mathsf{P/poly}$ if and only if $\mathsf{NEXP} = \mathsf{EXP} = \mathsf{MA}$. This result is crucial to Kabanets and Impagliazzo's breakthrough paper [14] showing that derandomizing BPP implies proving circuit lower bounds.

If we pay particular attention to MA, then the above separations and collapses match up nicely– if $\mathsf{EXP} \subseteq \mathsf{P/poly}$ then EXP collapses to MA, and if $\mathsf{EXP} \not\subseteq \mathsf{P/poly}$ then MA can be derandomized (and will be contained in $\mathsf{NSUBEXP}$).

The same is not true, however, for AM. Separating EXP from $\mathsf{NP/poly}$ implies that AM is contained in non-deterministic, sub-exponential time [21]. Placing $\mathsf{EXP} \subseteq \mathsf{NP/poly}$, however, implies only that $\mathsf{EXP} = \Sigma_3^{\mathsf{P}}$, the third level of the polynomial-time hierarchy[1].

[1] Actually one can prove that under the assumption that $\mathsf{EXP} \subseteq \mathsf{NP/poly}$, $\mathsf{EXP} \subseteq \mathsf{ZPP}^{\Sigma_2^{\mathsf{P}}}$ [6]

Is it true that $\mathsf{EXP} \subseteq \mathsf{NP}/\mathsf{poly}$ implies that $\mathsf{EXP} = \mathsf{AM}$? If so, combining this fact with the above derandomization of AM [21] would yield a rare unconditional derandomization of AM, namely that AM is contained in $\Sigma_2 - \mathsf{SUBEXP}$, the subexponential time analogue of Σ_2^{P} (AM is currently only known to be in Π_2^{P})– see Gutfreund et al. [11] for a discussion. Shaltiel and Umans [22] have asked if $\mathsf{EXP} \subseteq \mathsf{NP}/\log$ implies that $\mathsf{EXP} = \mathsf{AM}$, as even this is not known.

1.1 Our Results

We give a new collapse for exponential time if it is computed by a nondeterministic, slightly non-uniform complexity class. More precisely we show that if $\mathsf{EXP} \subseteq \mathsf{NP}/\log$ then $\mathsf{EXP} = \mathsf{P}_{||}^{\mathsf{NP}}$, i.e. EXP is computed by a polynomial-time turing machine with non-adaptive access to an NP-oracle. Further, we can also prove the converse:

Theorem 1 *The following are equivalent.*

1. $\mathsf{EXP} \subseteq \mathsf{P}_{||}^{\mathsf{NP}}$
2. $\mathsf{EXP} \subseteq \mathsf{NP}/\log$

The forward direction of our equivalence makes use of a new hardness amplification result due to Shaltiel and Umans. They prove that if $\mathsf{EXP} \not\subseteq \mathsf{NP}/\mathsf{poly}$ then $\mathsf{EXP} \not\subseteq \mathsf{P}_{||}^{\mathsf{NP}}/\mathsf{poly}$. The contrapositive gives a partial collapse of exponential time which we show how to strengthen via a non-standard method of computing advice. As a result we obtain $\mathsf{EXP} \subseteq \mathsf{P}_{||}^{\mathsf{NP}}$ implies $\mathsf{EXP} \subseteq \mathsf{NP}/\log$, improving on the conclusion $\mathsf{EXP} \subseteq \mathsf{AM}/\log$ obtained by Shaltiel and Umans [22].

The backwards direction requires two collapses. First we prove that if $\mathsf{EXP} \subseteq \mathsf{NP}/\log$ then $\mathsf{EXP} = \mathsf{P}^{\mathsf{NP}}$, and then we use the fact that the ODDMAXBIT function is complete for P^{NP} to show how the above advice strings can be computed and verified non-adaptively.

We also prove variations of Theorem 1 for other classes.

Theorem 2 *The following are equivalent.*

1. $\mathsf{PSPACE} \subseteq \mathsf{P}_{||}^{\mathsf{NP}}$
2. $\mathsf{PSPACE} \subseteq \mathsf{NP}/\log$

Theorem 3 *The following are equivalent.*

1. $\mathsf{P}^{\#\mathsf{P}} \subseteq \mathsf{P}_{||}^{\mathsf{NP}}$
2. $\mathsf{P}^{\#\mathsf{P}} \subseteq \mathsf{NP}/\log$

Is it possible to prove something similar to Theorem 1 for NEXP? We show that, in fact, such a statement is vacuously true for NEXP since one can separate NEXP from $\mathsf{NP}/\log$ outright via diagonalization (it is also known that $\mathsf{NEXP} \not\subseteq \mathsf{P}_{||}^{\mathsf{NP}}$ [10]). We can consider, however, the consequences of NEXP being contained in randomized complexity classes that take advice (such classes have been a focus of research interest as of late [4, 9]). We observe that the techniques of Impagliazzo et al. [12] can be used to prove that $\mathsf{NEXP} \subseteq \mathsf{BPP}/\log$ implies $\mathsf{NEXP} = \mathsf{BPP}$, strengthening a result of Trevisan and Vadhan [24].

1.2 Related Work

The first important collapse of a uniform class contained in a non-uniform class is due to Karp and Lipton [15] who showed that $\mathsf{NP} \subseteq \mathsf{P}/\mathsf{poly}$ implies that $\mathsf{PH} = \Sigma_2^{\mathsf{P}}$ and that $\mathsf{NP} \subset \mathsf{P}/\log$ implies $\mathsf{P} = \mathsf{NP}$. For exponential time, aside from the collapse results mentioned above due to Babai et al. [2] and Impagliazzo et al. [12], Buhrman and Homer [7] showed that if $\mathsf{EXP}^{\mathsf{NP}} \subseteq \mathsf{EXP}/\mathsf{poly}$ then $\mathsf{EXP}^{\mathsf{NP}} = \mathsf{EXP}$ and Buhrman, Fortnow, and Pavan [6] showed a weak relativization of Impagliazzo et al. [12], namely that for any $\mathsf{A} \in \mathsf{EXP}$, $\mathsf{NEXP}^{\mathsf{A}} \subseteq \mathsf{P}^{\mathsf{A}}/\mathsf{poly}$ implies $\mathsf{NEXP}^{\mathsf{A}} = \mathsf{EXP}^{\mathsf{A}}$ and if A is complete for Σ_k^{P} then $\mathsf{NEXP}^{\mathsf{A}} \subseteq \mathsf{P}^{\mathsf{A}}/\mathsf{poly}$ implies $\mathsf{NEXP}^{\mathsf{A}} = \mathsf{EXP} = \mathsf{MA}^{\mathsf{A}}$.

Buhrman, Chang and Fortnow [5] give an equivalence of a non-uniform collapse to NP and a uniform inclusion.

Theorem 4 (Buhrman-Chang-Fortnow) *The following are equivalent.*

1. $\mathsf{coNP} \subseteq \mathsf{NP}/1$
2. *The polynomial-time hierarchy collapses to* D^p

where D^p *is the set of languages that are the difference of two* NP *languages.*

Buhrman, Chang and Fortnow also generalize Theorem 4 to show that coNP in NP/k if and only if the polynomial-time hierarchy collapses to the 2^kth level of the Boolean hierarchy where the first level of the Boolean hierarchy is NP and the $i+1$st level is the set of differences of sets in NP and the sets in the ith level.

This extension only works for finite k but Buhrman, Fortnow and Chang conjecture that it extends to $k = O(\log n)$.

Conjecture 5 (Buhrman-Chang-Fortnow) *The following are equivalent.*

1. $\mathsf{coNP} \subseteq \mathsf{NP}/\log$
2. *The polynomial-time hierarchy collapses to* $\mathsf{P}_{||}^{\mathsf{NP}}$

Since EXP in $\mathsf{NP}/\mathsf{poly}$ implies $\mathsf{EXP} \subseteq \Sigma_3^{\mathsf{P}}$ [1, 26], Theorem 4 implies $\mathsf{EXP} \subseteq \mathsf{NP}/1$ if and only if $\mathsf{EXP} = \mathsf{D}^p$. Likewise Conjecture 5 implies Theorem 1 so we can view our Theorem 1 as a partial resolution of Conjecture 5.

2 Preliminaries

2.1 Complexity Classes

We assume the reader is familiar with complexity classes $\mathsf{P} = \cup_k \mathsf{DTIME}(n^k)$, $\mathsf{NP} = \cup_k \mathsf{NTIME}(n^k)$, $\mathsf{EXP} = \cup_k \mathsf{DTIME}(2^{n^k})$, $\mathsf{NEXP} = \cup_k \mathsf{NTIME}(2^{n^k})$, $\mathsf{PSPACE} = \cup_k \mathsf{DSPACE}(n^k)$ as well as notions of oracle turing machines and the polynomial-time hierarchy (see e.g. [3] for further explanations).

The non-uniform class $\mathsf{NP}/\log$ is the set of languages L such that there exists a language A in NP and a function $a : \mathcal{N} \to \Sigma^*$ with $|a(n)| = O(\log n)$ such that for all x in Σ^*, x is in L if and only if $(x, a(|x|))$ is in A. $\mathsf{NP/poly}$ has the same definition except that we allow $|a(n)| = O(n^k)$ for some k. Similarly NP can be replaced with any machine based complexity class, e.g. $\mathsf{BPP}/\log$ is the set of languages accepted by a BPP machine augmented with an advice string of length $O(\log n)$ which depends only on the input length.

The class P^{NP} consists of the languages accepted in polynomial-time with oracle access to some NP language. Since SAT, the set of satisfiable Boolean formula, is NP-complete, we can use SAT as the oracle language. We will make use of the following theorem giving a natural complete language for P^{NP}:

Theorem 6 (Krentel [17]) *Let $\phi(x_1, \ldots, x_n)$ be a Boolean formula. Let a be the lexicographically smallest satisfying assignment for ϕ, if there is one. The problem of determining whether the nth bit of a is equal to one is many-one complete for P^{NP}.*

The above language is often referred to as ODDMAXBIT.

The class $\mathsf{P}^{\mathsf{NP}}_{||}$ (sometimes written $\mathsf{P}^{\mathsf{NP}}_{tt}$) is the set of languages accepted in polynomial-time with non-adaptive oracle access to SAT, in other words all queries must be made before any the oracle returns any answers.

2.2 Randomized Classes

We also assume the reader is familiar with randomized complexity classes such as BPP and MA, the set of languages accepted by a Merlin-Arthur game where on input x, Merlin, the prover, sends a single message y and Arthur (the verifier) probabilistically verifies the purported proof y to determine membership of x. AM is the set of languages accepted by an Arthur-Merlin game where on input x, Arthur sends a random challenge r to Merlin who responds with y; Arthur then probabilistically verifies y to determine acceptance of x (see the survey by Kabanets [13]).

2.3 Alternation and Games

We will make use of the characterization of PSPACE due to Chandra, Kozen, and Stockmeyer as a game [8]. Chandra et al. showed that PSPACE is equivalent to the following two person game: on input x, players alternate announcing bits for a polynomial number of rounds and a polynomial-time computable judge chooses a winner based on x and the announced bits:

Theorem 7 (Chandra-Kozen-Stockmeyer) *A language L is in PSPACE if there exists a polynomial-time relation R on $2k+1$ strings where $k = n^{O(1)}$ and players P_1 and P_2 such that*

- *On round i for i odd, P_1 takes as input x and all strings from previous rounds and ouputs string x_i.*
- *On round j for j even, P_2 takes as input x and all strings from previous rounds and outputs string y_j.*
- *After k rounds, the input x is in the language L if and only if $R(x, x_1, y_1, x_2, y_2, \ldots, x_k, y_k)$ is true.*

Furthermore, each player P_i requires only PSPACE to output his/her string for each round. Hence we say each player has a *strategy* computable in PSPACE.

3 The Proof

In this section we give the proof of Theorem 1 showing the following are equivalent:

1. $\mathsf{EXP} \subset \mathsf{NP}/\mathsf{log}$
2. $\mathsf{EXP} \subset \mathsf{P}^{\mathsf{NP}}_{||}$

We use the following nice result of Shaltiel and Umans [22]:

Theorem 8 (Shaltiel-Umans) *If* $\mathsf{EXP} \subseteq \mathsf{P}^{\mathsf{NP}}_{||}/\mathsf{poly}$ *then* $\mathsf{EXP} \subseteq \mathsf{NP}/\mathsf{poly}$.

The proof of the above theorem makes use of the fact that EXP has a low-degree extension f, and if this extension is computable in $\mathsf{P}^{\mathsf{NP}}_{||}$ then for each query q made by the oracle-machine, one can give an advice p equal to the fraction of x's resulting in a $q(x)$ which should be answered as true by the NP oracle. For any x, it then suffices to choose a random low-degree curve through x and guess witnesses for a p fraction of points on this curve.

Proof of Theorem 1:

$(2 \Rightarrow 1)$

Fix an EXP-complete language L. By Theorem 8, L is in $\mathsf{NP}/\mathsf{poly}$. Fix the appropriate NP-machine M and let a_n be the lexicographically smallest advice string such that for all x of length n, x is in L iff $M(x, a_n)$ accepts.

Fix n. Let b_i be the ith bit of a_n. We can compute b_i in time exponential in n so by assumption b_i is in $\mathsf{P}^{\mathsf{NP}}_{||}$. Let Q_i be the set of queries to SAT made by the $\mathsf{P}^{\mathsf{NP}}_{||}$ algorithm to compute b_i. Let $Q = \bigcup_i Q_i$. Let r be the number of formulas in Q that are satisfiable. r is our $O(\log n)$ bits of advice.

Our $\mathsf{NP}/\log$ algorithm works as follows on input x of length n: guess a subset S of r formulas in Q and guess and verify their satisfying assignments. For each i, simulate the $\mathsf{P}^{\mathsf{NP}}_{||}$ algorithm to compute b_i answering each query yes if it is in S and no otherwise. From the b_i's we now have a_n. Now output $M(x, a_n)$.

$(1 \Rightarrow 2)$

This direction follows by combining the following two lemmas:

Lemma 9 *If* $\mathsf{EXP} \subseteq \mathsf{NP}/\log$ *then* $\mathsf{EXP} \subseteq \mathsf{P}^{\mathsf{NP}}$.

Lemma 10 *If* $\mathsf{P}^{\mathsf{NP}} \subseteq \mathsf{NP}/\log$ *then* $\mathsf{P}^{\mathsf{NP}} = \mathsf{P}^{\mathsf{NP}}_{||}$.

Proof of Lemma 9:

It is known that if EXP is in $\mathsf{NP/log}$ then $\mathsf{EXP} = \mathsf{PSPACE}$. This follows, for example, from the fact that if $\mathsf{EXP} \subset \mathsf{P^A/poly}$ then $\mathsf{EXP} \subseteq \mathsf{MA^A}$, i.e. a relativized version of a collapse due to Babai et al. observed by Buhrman et al. [2, 6]. Choosing $\mathsf{A} = \mathsf{NP}$ places $\mathsf{EXP} \subseteq \mathsf{MA^{NP}} \subseteq \mathsf{PSPACE}$.

By Theorem 7, we can view PSPACE as a interactive game between two players and a polynomial-time computable judge (recall each player's strategy is computable in PSPACE and thus $\mathsf{NP}/\log$ by assumption). Let L be a PSPACE-complete language and fix an input x. We will give an $\mathsf{P^{NP}}$ algorithm to determine whether x is in L.

Let T be the set of all $n^{O(1)}$ advice strings and let M be the NP advice taking machine deciding L. For each advice string $a \in T$, simulate $M(x, a)$ and divide T into two groups labeled IN and OUT depending on whether $M(x, a)$ accept or rejects. Since one advice string gives the correct answer, if either IN or OUT is empty then we know whether x is in L. This simulation can be carried out in $\mathsf{P^{NP}}$.

Otherwise, IN and OUT are both non-empty. Do the following for each pair of advice strings a_i and a_o where a_i is chosen from IN and a_o is chosen from OUT: simulate players P_1 and P_2 where P_1's strategy is computed using advice a_i and P_2's strategy is simulated using advice a_o. Since each strategy is in $\mathsf{PSPACE} \subseteq \mathsf{NP}/\log$, the entire simulation is computable in $\mathsf{P^{NP}}$.

Since some advice string a is the correct advice string, either a will be in IN and P_1 using this advice will defeat P_2 using any advice from OUT or vice versa. If the good advice string is in IN (and hence causes P_1 to always beat P_2), then we know x is in L and we will accept correctly. If we discover a to be in OUT we reject. ∎

Proof of Lemma 10:

From Theorem 6, we know that the ODDMAXBIT language consisting of the set of formulas whose lexicographically minimum satisfying assignment sets the last variable to true is complete for $\mathsf{P^{NP}}$. Hence, it suffices to give a $\mathsf{P^{NP}_{||}}$ algorithm for deciding ODDMAXBIT.

Given a formula ϕ of n variables, let a_i be the setting of the ith variable in the minimum satisfying assignment ($a_i = 0$ if there is no satisfying assignment). We can compute a_i in $\mathsf{P^{NP}}$ and thus in $\mathsf{NP}/\log$. Hence, given the correct advice we can compute a_i with one query to NP.

For each possible advice string b, we compute $a_1, \ldots, a_n$ via n parallel queries to NP (we can do this since each bit is computable by assumption by one independent query to NP). Given all of these purported minimum assignments, we find the lexicographically minimum assignment a' that satisfies ϕ. Since at least one advice is correct a' is the minimum satisfying assignment and the last bit of a' gives us the answer to the ODDMAXBIT question. ∎

3.1 Extending the Proof to PSPACE and $\mathsf{P^{\#P}}$

The proof of Theorem 8 in Shaltiel and Umans [22] extends to PSPACE and $\mathsf{P^{\#P}}$.

Theorem 11 (Shaltiel-Umans) *If* PSPACE *is in* $\mathsf{P^{NP}_{||}}$ *then* PSPACE *is in* $\mathsf{NP/poly}$. *If* $\mathsf{P^{\#P}}$ *is in* $\mathsf{P^{NP}_{||}}$ *then* $\mathsf{P^{\#P}}$ *is in* $\mathsf{NP/poly}$.

To prove Theorem 2 note that the proof of Theorem 1 goes through directly using PSPACE instead of EXP.

To prove Theorem 3 that $\mathsf{P}^{\#\mathsf{P}}$ is in $\mathsf{P}_{||}^{\mathsf{NP}}$ if and only if $\mathsf{P}^{\#\mathsf{P}}$ is in $\mathsf{NP}/\log$ we need a little more work. To show the "if" direction we first need the following lemma.

Lemma 12 *If $\mathsf{P}^{\#\mathsf{P}}$ is in $\mathsf{NP}/\mathsf{poly}$ then for every L in $\mathsf{P}^{\#\mathsf{P}}$ there exists an NP machine M and a sequence of advice strings $a_1, \ldots$ where*

1. *For all x, x is in L if and only if $M(x, a_{|x|})$ accepts,*
2. *For all n, $|a_n|$ is bounded by a fixed polynomial in n, and*
3. *The language $D = \{1^n 0^i \mid \text{the } i\text{th bit of } a_n \text{ is one}\}$ is in $\mathsf{P}^{\#\mathsf{P}}$.*

Proof:

Valiant [25] showed that Permanent (computing the ith bit of the permanent of a given 0-1 matrix) is Turing-complete for $\mathsf{P}^{\#\mathsf{P}}$. Similar to EXP, if the Permanent is in $\mathsf{P}^A/\mathsf{poly}$ then the Permanent is in MA^A [2, 6]. Setting $A = \text{SAT}$ we have Permanent in the polynomial-time hierarchy.

Let L be in $\mathsf{P}^{\#\mathsf{P}}$ and let M be an NP machine such that there exists a sequence of polynomially-long advice strings $b_1, \ldots$ where x in L if and only if $M(x, b_{|x|})$ accepts. Consider the language D consisting of the strings $1^n 0^i$ where the ith bit of the lexicographically least advice that computes L correctly on all inputs on length n is one. We can define D with a few quantifiers over L and L is reducible to the permanent which is in the polynomial-time hierarchy. This puts D in the polynomial-time hierarchy and thus in $\mathsf{P}^{\#\mathsf{P}}$ because of Toda's theorem [23] that every language in the polynomial-time hierarchy is in $\mathsf{P}^{\#\mathsf{P}}$. ▮

We can now prove that $\mathsf{P}^{\#\mathsf{P}}$ in $\mathsf{P}_{||}^{\mathsf{NP}}$ implies $\mathsf{P}^{\#\mathsf{P}}$ in $\mathsf{NP}/\log$ using the same techniques as the proof of Theorem 1 using Theorem 11 and Lemma 12.

Since $\mathsf{P}^{\mathsf{NP}} \subseteq \mathsf{P}^{\#\mathsf{P}}$, the other direction of Theorem 3 follows from the appropriate analog of Lemma 9.

Lemma 13 *If $\mathsf{P}^{\#\mathsf{P}} \subseteq \mathsf{NP}/\log$ then $\mathsf{P}^{\#\mathsf{P}} \subseteq \mathsf{P}^{\mathsf{NP}}$.*

Proof:

Fix a $\mathsf{P}^{\#\mathsf{P}}$ complete language L. If L is in $\mathsf{NP}/\log$ then L is in Σ_3^p, the third level of the polynomial-time hierarchy [18, 26]. We can view Σ_3^p as a three round game between two players and a polynomial-time judge. Each player's strategy is computable in the polynomial-time hierarchy and thus in $\mathsf{P}^{\#\mathsf{P}}$ by Toda [23]. We can now show L is in P^{NP} by the same argument as the proof of Lemma 9. ▮

4 On the Non-Uniform Complexity of NEXP

We would like to to extend the equivalence in Theorem 1 to hold for NEXP. We can do so, but the equivalence holds vacuously in the sense that NEXP is not contained in either class. Fu, Li and Zhong [10] showed that $\mathsf{NEXP} \not\subseteq \mathsf{P}_{||}^{\mathsf{NP}}$. This result and Theorem 1 does not immediately imply that

NEXP is not contained in $\mathsf{NP}/\log$ since we do not know how to directly show NEXP in $\mathsf{NP}/\log$ implies $\mathsf{NEXP} = \mathsf{EXP}$. Instead we prove the separation directly.

Theorem 14 $\mathsf{NEXP} \not\subset \mathsf{NP}/\log$.

Proof:

Assume by way of contradiction that $\mathsf{NEXP} \subseteq \mathsf{NP}/\log$. Then by a padding argument, $\mathsf{NEEXP} \subseteq \mathsf{NEXP}/\mathsf{poly}$. I.e. non-deterministic doubly exponential time is contained in a non-uniform analogue of NEXP. But now we apply the assumption that $\mathsf{NEXP} \subseteq \mathsf{NP}/\log$ again and obtain $\mathsf{NEEXP} \subseteq \mathsf{NP}/\mathsf{poly}$. Via a standard diagonalization argument one can show that even EEXP, deterministic doubly exponential time, does not have non-deterministic polynomial-size circuits. This is because in doubly exponential time we can enumerate over all say quasipolynomial-size non-deterministic circuits. ∎

4.1 NEXP versus randomized, non-uniform classes

In light of the fact that NEXP is known to not be in $\mathsf{NP}/\log$, it seems natural to consider the consequences of NEXP being contained in $\mathsf{BPP}/\log$ or $\mathsf{MA}/\log$. Separating NEXP from BPP is an outstanding open question; we prove this would also imply NEXP is not contained in $\mathsf{BPP}/\log$:

Theorem 15 $\mathsf{NEXP} \subseteq \mathsf{BPP}/\log$ *implies* $\mathsf{NEXP} = \mathsf{BPP}$.

The proof follows by combining two recent results from derandomization. The first is due to Impagliazzo et al. [12] who showed that $\mathsf{NEXP} \subset \mathsf{P}/\mathsf{poly}$ implies $\mathsf{NEXP} = \mathsf{MA}$. The second is due to Trevisan and Vadhan [24] who use the instance-checkability of EXP to show that $\mathsf{EXP} \subseteq \mathsf{BPP}/\log$ implies $\mathsf{EXP} \subseteq \mathsf{BPP}$. Theorem 15 follows by noticing that $\mathsf{NEXP} \subseteq \mathsf{BPP}/\log$ implies $\mathsf{NEXP} = \mathsf{EXP}$ (since $\mathsf{BPP} \subseteq \mathsf{P}/\mathsf{poly}$) and then applying the above result due to Trevisan and Vadhan [24].

5 Challenges

Is it possible to prove a similar consequence for NEXP and $\mathsf{MA}/\log$? Applying an argument from Impagliazzo et al. one can prove that $\mathsf{NEXP} \subseteq \mathsf{MA}/\log$ implies that either $\mathsf{NEXP} = \mathsf{EXP}$ or $\mathsf{NEXP} \subseteq \mathsf{NTIME}(2^{n^\epsilon})/n^\epsilon$. Unfortunately we do not know of a hierarchy theorem strong enough to show that the latter inclusion is false.

Another interesting avenue regarding NEXP would be to show that $\mathsf{NEXP} \subseteq \mathsf{P}_{||}^{\mathsf{NP}}/\log$ implies that $\mathsf{NEXP} = \mathsf{EXP}$.

Acknowledgements

Thanks to Chris Umans for providing us with an early copy of his latest work with Ronen Shaltiel [22] and verifying that Theorem 11 follows from that paper. We also thank Rahul Santhanam for interesting discussions and for the quick proof that $\mathsf{NEXP} \not\subseteq \mathsf{NP}/\log$.

References

[1] L. Babai, L. Fortnow, and C. Lund. Non-deterministic exponential time has two-prover interactive protocols. *Computational Complexity*, 1(1):3–40, 1991.

[2] L. Babai, L. Fortnow, N. Nisan, and A. Wigderson. BPP has subexponential simulations unless EXPTIME has publishable proofs. *Computational Complexity*, 3:307–318, 1993.

[3] J. Balcázar, J. Díaz, and J. Gabarró. *Structural Complexity I.* Springer, 1988.

[4] B. Barak. A probabilistic-time hierarchy theorem for "Slightly Non-uniform" algorithms. In *Proceedings of Randomization and Approximation Techniques: 6th International Workshop*, volume 2483, pages 194–208. Springer, Berlin, 2002.

[5] H. Buhrman, R. Chang, and L. Fortnow. One bit of advice. In *Proceedings of the 20th Symposium on Theoretical Aspects of Computer Science*, volume 2607 of *Lecture Notes in Computer Science*, pages 547–558. Springer, Berlin, 2003.

[6] H. Buhrman, L. Fortnow, and A. Pavan. Some results on derandomization. In *Proceedings of the 20th Symposium on Theoretical Aspects of Computer Science*, volume 2607 of *Lecture Notes in Computer Science*, pages 212–222. Springer, Berlin, 2003.

[7] H. Buhrman and S. Homer. Superpolynomial circuits, almost sparse oracles and the exponential hierarchy. In *Proceedings of the 12th Conference on the Foundations of Software Technology and Theoretical Computer Science*, volume 652 of *Lecture Notes in Computer Science*, pages 116–127. Springer, Berlin, Germany, 1992.

[8] A. Chandra, D. Kozen, and L. Stockmeyer. Alternation. *Journal of the ACM*, 28(1):114–133, 1981.

[9] L. Fortnow and R. Santhanam. Hierarchy theorems for probabilistic polynomial time. In *Proceedings of the 45th IEEE Symposium on Foundations of Computer Science*, pages 316–324. IEEE, New York, 2004.

[10] B. Fu, H. Li, and Y. Zhong. An application of the translational method. *Mathematical Systems Theory*, 27:183–186, 1994.

[11] D. Gutfreund, R. Shaltiel, and A. Ta-Shma. Uniform hardness vs. randomness tradeoffs for arthur-merlin games. In *Proceedings of the 18th IEEE Conference on Computational Complexity*, pages 33–47. IEEE, New York, 2003.

[12] R. Impagliazzo, V. Kabanets, and A. Wigderson. In search of an easy witness: Exponential versus probabilistic time. *Journal of Computer and System Sciences*, 65(4):672–694, 2002.

[13] V. Kabanets. Derandomization: A brief overview. *Bulletin of the European Association for Theoretical Computer Science*, 76, February 2002.

[14] V. Kabanets and R. Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. In *Proceedings of the 35th ACM Symposium on the Theory of Computing*, pages 355–364, New York, 2003. ACM.

[15] R. Karp and R. Lipton. Some connections between nonuniform and uniform complexity classes. In *Proceedings of the 12th ACM Symposium on the Theory of Computing*, pages 302–309. ACM, New York, 1980.

[16] A. Klivans and D. van Melkebeek. Graph nonisomorhism has subexponential size proofs unless the polynomial-time hierarchy collapses. *SIAM Journal on Computing*, 31(5):1501–1526, 2002.

[17] M. Krentel. The complexity of optimization problems. *Journal of Computer and System Sciences*, 36(3):490–509, June 1988.

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

[19] P. Miltersen and V. Vinodchandran. Derandomizing Arthur-Merlin games using hitting sets. In *Proceedings of the 40th IEEE Symposium on Foundations of Computer Science*, pages 71–80. IEEE, New York, 1999.

[20] N. Nisan and A. Wigderson. Hardness vs. randomness. *Journal of Computer and System Sciences*, 49:149–167, 1994.

[21] R. Shaltiel and C. Umans. Simple extractors for all min-entropies and a new pseudo-random generator. In IEEE, editor, *42nd IEEE Symposium on Foundations of Computer Science: proceedings: October 14–17, 2001, Las Vegas, Nevada, USA*, pages 648–657, 1109 Spring Street, Suite 300, Silver Spring, MD 20910, USA, 2001. IEEE Computer Society Press.

[22] R. Shaltiel and C. Umans. Pseudorandomness for approximate counting and sampling. In *Electronic Colloquium on Computational Complexity*, number 04-086. 2004.

[23] S. Toda. PP is as hard as the polynomial-time hierarchy. *SIAM Journal on Computing*, 20(5):865–877, 1991.

[24] L. Trevisan and S. Vadhan. Pseudorandomness and average-case complexity via uniform reductions. In *Proceedings of the 17th IEEE Conference on Computational Complexity*, pages 103–112. IEEE, New York, 2002.

[25] L. Valiant. The complexity of computing the permanent. *Theoretical Computer Science*, 8:189–201, 1979.

[26] C. Yap. Some consequences of nonuniform conditions on uniform classes. *Theoretical Computer Science*, 26:287–300, 1983.