Derandomizing Arthur-Merlin Games

Dieter van Melkebeek*
The University of Chicago

July 14, 1998

Abstract

We establish hardness versus randomness trade-offs for Arthur-Merlin games. We create efficient nondeterministic simulations of bounded round Arthur-Merlin games, using a language in exponential time which small circuits cannot decide given access to an oracle for satisfiability. Our results yield subexponential size proofs for graph nonisomorphism at infinitely many lengths unless the polynomial-time hierarchy collapses. The same holds for any other language with a bounded round interactive proof system. We also apply our techniques to related complexity classes.

1 Introduction

Randomness helps computer scientists considerably in designing algorithms. Several computational problems seem easier to solve when we can flip coins. From a complexity theoretic point of view though, the power of random bits remains widely open: Can they reduce the time or space we need to compute solutions? As of now, we do not know the answer to such fundamental questions about randomness.

However, a major insight in complexity theory states that if some of the seemingly hard algorithmic problems, really are computationally intractable, then randomness cannot help much. Nisan and Wigderson [NW94] established a range of hardness versus randomness trade-offs. They showed how to use a language in exponential time that is non-uniformly hard in an average-case sense, to construct a pseudo-random generator that fools circuits of polynomial size. This way, they obtained nontrivial derandomizations of polynomial-time randomized algorithms under average-case hardness assumptions, and even deterministic polynomial-time simulations under the strongest of their hypotheses. Babai, Fortnow, Nisan, and Wigderson [BFNW93] and Impagliazzo and Wigderson [IW97] relaxed the hardness condition from average-case to worst-case. As a corollary, we can simulate every polynomial-time randomized algorithm deterministically in subexponential time for infinitely many input lengths unless the polynomial-time hierarchy collapses [BFNW93].

Interactive proof checking constitutes another area where coin flips seem beneficial. Whereas in the absence of randomness, interaction with an all-powerful prover allows us to efficiently check membership for languages in NP, in the presence of randomness we can do it for all of PSPACE

*Research partially supported by the NSF through grant CCR 92-53582, by the European Union through TMR grant ERB-4001-GT-96-0783 while visiting CWI and the University of Amsterdam, and by the Fields Institute while visiting the Fields Institute at the University of Toronto.

[LFKN92, Sha92]. Even in a bounded number of rounds of randomized interaction, it looks like we can recognize more than NP. Graph nonisomorphism forms a prototypical example. We know a constant round interactive proof system for it [GMW91], but no membership proofs of polynomial size. Even the existence of subexponential size proofs remains open until today. Goldwasser and Sipser [GS89] established an equivalence between bounded round interactive proof systems and bounded round Arthur-Merlin games. Babai [Bab85, BM88] introduced the latter and showed how to transform any bounded round game into an equivalent AM-game, a two round game where Arthur moves first.

In this paper, we establish randomness versus hardness trade-offs for Arthur-Merlin games, similar to the ones for algorithms. Our approach works for other related classes as well, but yields the most interesting results for the class AM of languages with bounded round interactive proof systems. We show that the existence of an exponential-time decidable language with high worst-case nonuniform complexity when the circuits have access to an oracle for satisfiability, implies nontrivial derandomizations of AM-games. The trade-offs are presented in Table 1. We use C^B to denote circuit complexity given access to oracle B, and similarly $\tilde{C}^B$ given only parallel access to the oracle. See Section 2 for precise definitions. The parameter s in Table 1 can be any space constructible function, the interesting range lying between logarithmic and subpolynomial, e.g., polylogarithmic.

hardness assumption:	derandomization consequence:
$\exists\, f \in \mathrm{NEXP} \cap \mathrm{coNEXP} : \tilde{C}_f^{\mathrm{SAT}}(n) \in n^{\omega(1)}$	$\mathrm{AM} \subseteq \cap_{\epsilon>0}\mathrm{NTIME}[2^{n^\epsilon}]$
$\exists\, f \in \mathrm{NEXP} \cap \mathrm{coNEXP} : \tilde{C}_f^{\mathrm{SAT}}(s(n)) \in \Omega(n)$	$\mathrm{AM} \subseteq \cup_{c>0}\mathrm{NTIME}[2^{(s(n^c))^{O(1)}}]$
$\exists\, f \in \mathrm{NE} \cap \mathrm{coNE} : \tilde{C}_f^{\mathrm{SAT}}(n) \in 2^{\Omega(n)}$	$\mathrm{AM} = \mathrm{NP}$

Table 1: Hardness versus randomness trade-offs for AM.

If the hardness condition on the left-hand side of Table 1 holds for infinitely many input lengths, then the corresponding derandomization on the right-hand side works for infinitely many input lengths. Similar weak interpretations hold for all our results. As a corollary to the weak version of Table 1, we obtain that every language in AM, and graph nonisomorphism in particular, has subexponential size proofs for infinitely many input sizes unless the polynomial-time hierarchy collapses.

Using other hardness measures and various oracles B, we get derandomization results for other complexity classes. We summarize the situation in Table 2.

lower bound on:	derandomizes:
C	MA
$\tilde{C}^{\mathrm{SAT}}$	AM
C^B	BPP^B

Table 2: Overview of derandomization results.

The rest of this paper is organized as follows. After introducing our notation in Section 2, we establish our theorems for AM in Section 3. Section 4 discusses our results for other complexity classes, and the last section presents an open problem relevant to derandomizing AM.

2 Notation

Most of our complexity theoretic notation is standard. We refer the reader to the textbooks by Balcázar, Díaz and Gabarró [BDG95, BDG90], and by Papadimitriou [Pap94].

An *oracle circuit* D is a circuit with AND, OR and NOT gates as well as oracle gates, which compute membership of the string formed by their input bits to some unspecified oracle B. The function D^B the circuit computes depends on the oracle B. A *parallel oracle circuit* is an oracle circuit in which every path contains at most one oracle gate. Equivalently, a parallel oracle circuit makes all its queries at the same level. In this paper, we measure the *size* of a circuit by its number of connections.

Given a Boolean function $f : \{0,1\}^* \to \{0,1\}$ and an oracle B, the *circuit complexity* $C_f^B(n)$ of f at length n relative to B is the smallest integer t such that there is an oracle circuit of size t that computes f on inputs of length n when given B as an oracle. The *hardness* $H_f^B(n)$ of f at length n relative to B is the largest integer t such that for any oracle circuit D of size at most t with n inputs

$$|\Pr_x[D^B(x) = f(x)] - \frac{1}{2}| \leqslant \frac{1}{t},$$

where x is uniformly distributed over $\{0,1\}^n$.

Similarly, using parallel oracle circuits instead of oracle circuits, we define $\tilde{C}_f^B(n)$ and $\tilde{H}_f^B(n)$. The classical circuit complexity of f at length n equals $C_f^{\varnothing}(n)$ and is also denoted by $C_f(n)$.

A *pseudo-random generator* G is a sequence of functions $(G_n)_n$ such that G_n maps $\{0,1\}^{s(n)}$ to $\{0,1\}^n$ for some function $s : \mathbb{N} \to \mathbb{N}$ with $s(n) < n$. The function s is called the *seed length* of G. We say that G is *computable in* $\mathcal{C}$ if the problem of deciding the i-th bit of $G_n(\sigma)$ given $\langle n, \sigma, i\rangle$ belongs to $\mathcal{C}$. Given an oracle B, G is called *secure against* B if for almost all n and for any oracle circuit D of size at most n

$$|\Pr_\rho[D^B(\rho) = 1] - \Pr_\sigma[D^B(G_n(\sigma)) = 1]| \leqslant \frac{1}{n}, \tag{1}$$

where ρ is uniformly distributed over $\{0,1\}^n$ and σ over $\{0,1\}^{s(n)}$.

Similarly, if (1) holds for *parallel* oracle circuits D, we say that G is *secure against parallel* B. In the case where $B = \varnothing$, we call G *secure*.

For any function $t : \mathbb{N} \to \mathbb{N}$, AM-TIME[$t(n)$] represents the class of languages L for which there exists a deterministic Turing machine M that runs in time $O(t(n))$ on inputs of the form $\langle x, y, z\rangle$ where $x \in \{0,1\}^n$ and $y, z \in \{0,1\}^{t(n)}$, such that for any input x,

$$x \in L \quad \Rightarrow \quad \Pr_{|y|=t(n)}[\exists z \in \{0,1\}^{t(n)} : M(\langle x,y,z\rangle) = 1] \geqslant \frac{2}{3} \tag{2}$$

$$x \notin L \quad \Rightarrow \quad \Pr_{|y|=t(n)}[\exists z \in \{0,1\}^{t(n)} : M(\langle x,y,z\rangle) = 1] \leqslant \frac{1}{3}, \tag{3}$$

where $n = |x|$ and the probabilities are with respect to the uniform distribution. AM denotes $\cup_{c>0}$AM-TIME[n^c].

Similarly, MA-TIME[$t(n)$] equals the class of languages L for which there exists a deterministic Turing machine M that runs in time $O(t(n))$ on inputs of the form $\langle x, y, z\rangle$ where $x \in \{0,1\}^n$ and

$y, z \in \{0,1\}^{t(n)}$, such that for any input x,

$$x \in L \;\Rightarrow\; \exists\, y \in \{0,1\}^{t(n)} : \Pr_{|z|=t(n)}[M(\langle x,y,z\rangle) = 1] \geqslant \frac{2}{3}$$
$$x \notin L \;\Rightarrow\; \forall\, y \in \{0,1\}^{t(n)} : \Pr_{|z|=t(n)}[M(\langle x,y,z\rangle) = 1] \leqslant \frac{1}{3}.$$

MA is a shortcut for $\cup_{c>0}$MA-TIME$[n^c]$.

EXP $= \cup_{c>0}$DTIME$[2^{n^c}]$ and E $= \cup_{c>0}$DTIME$[2^{cn}]$. Similarly, NEXP $= \cup_{c>0}$NTIME$[2^{n^c}]$ and NE $= \cup_{c>0}$NTIME$[2^{cn}]$.

For any complexity class $\mathcal{C}$, the class i.o.-$\mathcal{C}$ consists of all languages L for which there is a language $L' \in \mathcal{C}$ such that $L \cap \{0,1\}^n = L' \cap \{0,1\}^n$ for infinitely many lengths n.

For any two languages A and B, $A \oplus B$ denotes the language $\{0x \mid x \in A\} \cup \{1x \mid x \in B\}$.

For any function $s : \mathbb{N} \to \mathbb{N}$, $\Omega(s)$ denotes the class of all functions $t : \mathbb{N} \to \mathbb{N}$ for which there exists a constant $\epsilon > 0$ such that $t(n) \geqslant \epsilon \cdot s(n)$ for almost all n. $\omega(s)$ is the class of t such that for any $c > 0$ and almost all n, $t(n) \geqslant c \cdot s(n)$.

3 Derandomizing AM

Our derandomization results for AM $= \cup_{c>0}$AM-TIME$[n^c]$ are based on the following lemma, which states that it suffices to construct efficient pseudo-random generators that are secure against parallel SAT.

Lemma 3.1 *Let $s : \mathbb{N} \to \mathbb{N}$ be a space constructible function, and $t, \tau : \mathbb{N} \to \mathbb{N}$ be time constructible functions. If there is a pseudo-random generator G computable in* NTIME$[\tau(n)] \cap$ coNTIME$[\tau(n)]$ *and with seed length s that is secure against parallel* SAT*, then*

$$\text{AM-TIME}[t(n)] \subseteq \text{NTIME}[2^{s(t'(n))} \cdot \tau(s(t'(n))) \cdot t(n)], \tag{4}$$

where $t'(n) \in O(t(n) \log^2 t(n))$.

We give the proof in the Appendix. We can also establish the variant of Lemma 3.1 in which we weaken the hypothesis and only require the security condition (1) to hold for infinitely many n, and weaken the conclusion (4) to

$$\text{AM-TIME}[t(n)] \subseteq \text{i.o.-NTIME}[2^{s(t'(n))} \cdot \tau(s(t'(n))) \cdot t(n)].$$

All our results come in these two versions. We refer to the form given in Lemma 3.1 as the strong version, and to the latter variant as the weak version. We will typically only state one of the versions.

We can construct pseudo-random generators secure against parallel B from functions which are hard to compute for circuits with parallel oracle access to B.

Theorem 3.2 *Let A be a class of oracles, B an oracle, and $s : \mathbb{N} \to \mathbb{N}$ a space constructible function. For any Boolean function $f \in$ EXP^A, there is a pseudo-random generator G computable in* EXP^A *which is secure against parallel B and has seed length s provided*

$$\tilde{C}_f^B\left(\sqrt[6]{s(n)}\right) \in \omega(n^9).$$

Proof
Nisan and Wigderson used relativizable techniques to show the following theorem.

Theorem 3.3 ([NW94]) *Let A be a class of oracles, B an oracle, and $s : \mathbb{N} \to \mathbb{N}$ a space constructible function. For any Boolean function $g \in \mathrm{EXP}^A$, there is a pseudo-random generator G computable in EXP^A which is secure against parallel B and has seed length s provided*

$$\tilde{H}_g^B(\sqrt{s(n)}) \geqslant n^2.$$

Babai, Fortnow, Nisan, and Wigderson gave a relativizable proof of the next result.

Theorem 3.4 ([BFNW93]) *Let A be a class of oracles and B an oracle. For any Boolean function $f \in \mathrm{EXP}^A$, there is a Boolean function $g \in \mathrm{EXP}^A$ such that*

$$\tilde{H}_g^B(n) \in \Omega\left(\sqrt[3]{\tilde{C}_f^B(\sqrt[3]{n})}/n^2\right).$$

Combining Theorems 3.3 and 3.4 finishes the proof of Theorem 3.2. □

In this section, we will apply Theorem 3.2 with $A = \mathrm{NP} \cap \mathrm{coNP}$ and $B = \mathrm{SAT}$. Note that $\mathrm{EXP}^{\mathrm{NP}\cap\mathrm{coNP}} = \mathrm{NEXP} \cap \mathrm{coNEXP}$. Together with Lemma 3.1, Theorem 3.2 yields the following results.

Theorem 3.5 *If there is a Boolean function $f \in \mathrm{NEXP} \cap \mathrm{coNEXP}$ such that $\tilde{C}_f^{\mathrm{SAT}}(n) \in n^{\omega(1)}$, then*

$$\mathrm{AM} \subseteq \cap_{\epsilon>0}\mathrm{NTIME}[2^{n^\epsilon}].$$

Theorem 3.6 *If there is a Boolean function $f \in \mathrm{NEXP} \cap \mathrm{coNEXP}$, and a space constructible function $s : \mathbb{N} \to \mathbb{N}$ such that*

$$\tilde{C}_f^{\mathrm{SAT}}(s(n)) \in \Omega(n),$$

then there is a constant d such that

$$\mathrm{AM} \subseteq \cup_{c>0}\mathrm{NTIME}[2^{(s(n^c))^d}].$$

We can rephrase the weak version of Theorem 3.5 as:

Theorem 3.7 *If $\mathrm{NEXP} \cap \mathrm{coNEXP} \not\subseteq \mathrm{P}_{||}^{\mathrm{NP}}/\mathrm{poly}$, then $\mathrm{AM} \subseteq \cap_{\epsilon>0}\text{i.o.-}\mathrm{NTIME}[2^{n^\epsilon}]$.*

So, if the conclusion of Theorem 3.7 fails to hold, then $\mathrm{NEXP} \cap \mathrm{coNEXP} \subseteq \mathrm{P}_{||}^{\mathrm{NP}}/\mathrm{poly}$, which implies that $\mathrm{EXP} = \Sigma_3^{\mathrm{p}} \cap \Pi_3^{\mathrm{p}}$ and so that the polynomial-time hierarchy collapses to the third level. Therefore, we obtain:

Corollary 3.8 *Every language in AM, and graph nonisomorphism in particular, has subexponential size proofs for infinitely many lengths unless the polynomial-time hierarchy collapses.*

Theorem 3.6 yields a range of hardness versus randomness trade-offs for the various choices of the parameter s. Since $C_f^B(n) \in O(2^n/n)$ always holds, the hypothesis of Theorem 3.6 cannot be met for s sublogarithmic. On the other hand, the conclusion becomes trivial in the case where s is polynomial. Therefore, the interesting range for s lies between logarithmic and subpolynomial. For example, for s polylogarithmic, Theorem 3.6 reads:

Theorem 3.9 *Every language in* AM *has quasipolynomial size proofs unless all of* NEXP$\cap$coNEXP *has subexponential size circuits for infinitely many lengths when given parallel oracle access to* SAT.

Theorem 3.6 is not powerful enough to yield a complete derandomization of AM: Even if s is logarithmic, it only puts AM in nondeterministic quasipolynomial time. However, we can strengthen Theorem 3.6 for logarithmic s as follows.

Theorem 3.10 *If there is a Boolean function* $f \in \mathrm{NE} \cap \mathrm{coNE}$ *such that* $\tilde{C}_f^{\mathrm{SAT}}(n) \in 2^{\Omega(n)}$, *then* $\mathrm{AM} = \mathrm{NP}$.

The proof follows from Lemma 3.1 and the next theorem applied to $A = \mathrm{NP} \cap \mathrm{coNP}$ and $B = \mathrm{SAT}$, noting that $\mathrm{E}^{\mathrm{NP}\cap\mathrm{coNP}} = \mathrm{NE} \cap \mathrm{coNE}$.

Theorem 3.11 *Let A be a class of oracles and B an oracle. If there is Boolean function* $f \in \mathrm{E}^A$ *such that* $\tilde{C}_f^B(n) \in 2^{\Omega(n)}$, *then there is a pseudo-random generator G computable in* E^A *which is secure against parallel B and has seed length* $O(\log n)$.

Proof
Nisan and Wigderson showed the following theorem using relativizable techniques.

Theorem 3.12 ([NW94]) *Let A be a class of oracles and B an oracle. If there is a Boolean function* $g \in \mathrm{E}^A$ *with* $\tilde{H}_g^B(n) \in 2^{\Omega(n)}$, *then there is a pseudo-random generator G computable in* E^A *which is secure against parallel B and has seed length* $O(\log n)$.

Impagliazzo and Wigderson gave a relativizable proof of:

Theorem 3.13 ([IW97]) *Let A be a class of oracles and B an oracle. For any Boolean function* $f \in \mathrm{E}^A$ *such that* $\tilde{C}_f^B(n) \in 2^{\Omega(n)}$, *there is a Boolean function* $g \in \mathrm{E}^A$ *such that* $\tilde{H}_g^B(n) \in 2^{\Omega(n)}$.

Combining Theorems 3.12 and 3.13 finishes the proof of Theorem 3.11. $\square$

4 Derandomizing Classes Other than AM

The approach of Section 3 also applies to certain other classes than AM. By modifying the hardness measure and choosing various oracles B, we obtain results for MA and BPP^B, as indicated in Table 2.

For derandomizing $\mathrm{MA} = \cup_{c>0}\text{MA-TIME}[n^c]$, we do not need oracle access at all: An efficient secure pseudo-random generator suffices.

Lemma 4.1 *Let* $s : \mathbb{N} \to \mathbb{N}$ *be a space constructible function, and* $t, \tau : \mathbb{N} \to \mathbb{N}$ *be time constructible functions. If there is a secure pseudo-random generator G computable in* $\mathrm{NTIME}[\tau(n)] \cap \mathrm{coNTIME}[\tau(n)]$ *and with seed length s, then*

$$\text{MA-TIME}[t(n)] \subseteq \mathrm{NTIME}[2^{s(t'(n))} \cdot \tau(s(t'(n))) \cdot t(n)],$$

where $t'(n) \in O(t(n) \log t(n))$.

The unrelativized versions of Theorems 3.2 and 3.11 yield results analogous to Theorems 3.5, 3.6, 3.9, and 3.10, presented in Table 3.

The analogue of Theorem 3.7 reads:

hardness assumption:	derandomization consequence:
$\exists\, f \in \mathrm{NEXP} \cap \mathrm{coNEXP} : C_f(n) \in n^{\omega(1)}$	$\mathrm{MA} \subseteq \cap_{\epsilon>0}\mathrm{NTIME}[2^{n^\epsilon}]$
$\exists\, f \in \mathrm{NEXP} \cap \mathrm{coNEXP} : C_f(s(n)) \in \Omega(n)$	$\mathrm{MA} \subseteq \cup_{c>0}\mathrm{NTIME}[2^{(s(n^c))^{O(1)}}]$
$\exists\, f \in \mathrm{NE} \cap \mathrm{coNE} : C_f(n) \in 2^{\Omega(n)}$	$\mathrm{MA} = \mathrm{NP}$

Table 3: Hardness versus randomness trade-offs for MA.

Theorem 4.2 *If* $\mathrm{NEXP} \cap \mathrm{coNEXP} \not\subseteq \mathrm{P/poly}$*, then* $\mathrm{MA} \subseteq \cap_{\epsilon>0}\text{i.o.-}\mathrm{NTIME}[2^{n^\epsilon}]$.

Consequently, if MA does not have subexponential size proofs for infinitely many lengths, then $\mathrm{NEXP} \cap \mathrm{coNEXP} \subseteq \mathrm{P/poly}$, which implies that $\mathrm{EXP} = \Sigma_2^{\mathrm{P}} \cap \Pi_2^{\mathrm{P}}$, hence that the polynomial-time hierarchy collapses to the second level.

Finally, by looking at oracle circuits instead of parallel oracle circuits, we can derandomize BPP^B for any oracle B, given an exponential-time computable predicate with high circuit complexity relative to B. Replacing $\tilde{C}$ by C, $\tilde{H}$ by H, and dropping the "parallel" in Theorems 3.2 and 3.11 and their proofs, yields the derandomization results of Table 4. Both A and B represent arbitrary oracles.

hardness assumption:	derandomization consequence:
$\exists\, f \in \mathrm{EXP}^A : C_f^B(n) \in n^{\omega(1)}$	$\mathrm{BPP}^B \subseteq \cap_{\epsilon>0}\mathrm{DTIME}^{A\oplus B}[2^{n^\epsilon}]$
$\exists\, f \in \mathrm{EXP}^A : C_f^B(s(n)) \in \Omega(n)$	$\mathrm{BPP}^B \subseteq \cup_{c>0}\mathrm{DTIME}^{A\oplus B}[2^{(s(n^c))^{O(1)}}]$
$\exists\, f \in \mathrm{E}^A : C_f^B(n) \in 2^{\Omega(n)}$	$\mathrm{BPP}^B \subseteq \mathrm{P}^{A\oplus B}$

Table 4: Hardness versus randomness trade-offs for BPP^B.

We note that in the simulations on the right-hand side of Table 4, the queries to B are always polynomially bounded in size.

The analogue of Theorem 3.7 follows.

Theorem 4.3 *Let A and B be oracles. If* $\mathrm{EXP}^A \not\subseteq \mathrm{P}^B/\mathrm{poly}$*, then* $\mathrm{BPP}^B \subseteq \cap_{\epsilon>0}\text{i.o.-}\mathrm{DTIME}^{A\oplus B}[2^{n^\epsilon}]$.

Setting $B = \mathrm{SAT}$ and $A = \mathrm{SAT}$ or $A = \varnothing$ respectively, we obtain:

Corollary 4.4 *If* $\mathrm{EXP}^{\mathrm{NP}} \not\subseteq \mathrm{P}^{\mathrm{NP}}/\mathrm{poly}$*, then* $\mathrm{BPP}^{\mathrm{NP}} \subseteq \cap_{\epsilon>0}\text{i.o.-}\mathrm{DTIME}^{\mathrm{NP}}[2^{n^\epsilon}]$*. If* $\mathrm{EXP} \not\subseteq \mathrm{P}^{\mathrm{NP}}/\mathrm{poly}$*, then* $\mathrm{BPP}^{\mathrm{NP}} \subseteq \cap_{\epsilon>0}\text{i.o.-}\mathrm{DTIME}^{\mathrm{NP}}[2^{n^\epsilon}]$*, where the queries to the* NP*-oracle are polynomially bounded in size.*

Note that $\mathrm{BPP}^{\mathrm{NP}}$ contains AM.

5 Further Research

In Section 3, we showed that efficient pseudo-random generators secure against SAT allow us to derandomize AM. In fact, it suffices for the pseudo-random generator to be secure against nondeterminism.

Definition 5.1 *A* nondeterministic circuit *D is a circuit that has an extra input y in addition to its real input x. $D(x)$ equals 1 if there is a setting of y that makes the circuit accept, and 0 otherwise. A pseudo-random generator G is* secure against nondeterminism *if for almost all lengths n and any nondeterministic circuit D of size at most n*

$$|\Pr_{\rho}[D(\rho) = 1] - \Pr_{\sigma}[D(G_n(\sigma)) = 1]| \leqslant \frac{1}{n},$$

where ρ is uniformly distributed over $\{0,1\}^n$ and σ over $\{0,1\}^{s(n)}$, and s denotes the seed length of G.

The proof of Lemma 3.1 also shows the following.

Lemma 5.1 *Let $s : \mathbb{N} \to \mathbb{N}$ be a space constructible function, and $t, \tau : \mathbb{N} \to \mathbb{N}$ be time constructible functions. If there is a pseudo-random generator G computable in* $\mathrm{NTIME}[\tau(n)] \cap \mathrm{coNTIME}[\tau(n)]$ *and with seed length s that is secure against nondeterminism, then*

$$\text{AM-TIME}[t(n)] \subseteq \mathrm{NTIME}[2^{s(t'(n))} \cdot \tau(s(t'(n))) \cdot t(n)],$$

where $t'(n) \in O(t(n) \log t(n))$.

We can define the *nondeterministic circuit complexity* and the *nondeterministic hardness* of a Boolean function f similar to C_f^B respectively H_f^B from Section 2, using nondeterministic circuits instead of oracle circuits.

The constructions of efficient pseudo-random generators out of a Boolean function with high average-case complexity as given by Theorems 3.3 and 3.12, carry over to the nondeterministic setting. However, the transformations of a function with high worst-case complexity into one with high average-case complexity as in Theorems 3.4 and 3.13, seem problematic. The latter involve two steps:

1. Creating a Boolean function which is mildly hard to compute on average.

2. Boosting the average-case hardness to the required level.

The second step forms the bottleneck. Its key ingredient is Yao's XOR Lemma [Yao82], which states that taking the exclusive OR of a Boolean function applied to independent arguments of the same length increases the average-case hardness. We refer the reader to the paper by Goldreich, Nisan and Wigderson [GNW95] for a survey of quantitative statements of the XOR Lemma. Known proofs [Lev87, BH89, Imp95, GNW95, IW97] break down for nondeterministic circuits.

Open Problem 5.1 *Does the XOR Lemma hold for nondeterministic circuits?*

Positive answers would allow us to extend Theorems 3.4 and 3.13 to the nondeterministic setting, and thus relax the hardness assumptions in Table 1 from "circuit complexity for parallel access to SAT" to "nondeterministic circuit complexity".

Acknowledgments

I want to express my sincere gratitude to Oded Goldreich, Adam Klivans, Madhu Sudan, Luca Trevisan, and Salil Vadhan for their strong encouragement and helpful collaboration. Without them, this write-up would not exist.

I would also like to thank the following people for interesting discussions about this paper in general and the open problem in particular: Behfar Bastani, Lance Fortnow, Valentine Kabanets, Mike Saks, Avi Wigderson, and Shiyu Zhou.

I am grateful to Lance Fortnow and Valentine Kabanets for their comments on the write-up.

References

[Bab85] L. Babai. Trading group theory for randomness. In *Proceedings of the 17th ACM Symposium on the Theory of Computing*, pages 421–429, New York, 1985. ACM.

[BDG90] J. Balcázar, J. Díaz, and J. Gabarró. *Structural Complexity II*, volume 22 of *EATCS Monographs on Theoretical Computer Science*. Springer-Verlag, 1990.

[BDG95] J. Balcázar, J. Díaz, and J. Gabarró. *Structural Complexity I*, volume 11 of *EATCS Monographs on Theoretical Computer Science*. Springer-Verlag, 1995.

[BFNW93] L. Babai, L. Fortnow, N. Nisan, and A. Wigderson. BPP has subexponential time simulations unless EXPTIME has publishable proofs. *Computational Complexity*, 3:307–318, 1993.

[BH89] R. Boppana and R. Hirschfeld. Pseudorandom generators and complexity classes. In S. Micali, editor, *Randomness and Computations*, volume 5 of *Advances in Computing Research*, pages 1–26. JAI Press, Greenwich, 1989.

[BM88] L. Babai and S. Moran. Arthur-Merlin games: a randomized proof system, and a hierarchy of complexity classes. *Journal of Computer and System Sciences*, 36:254–276, 1988.

[Coo88] S. Cook. Short propositional formulas represent nondeterministic computations. *Information Processing Letters*, 26:269–270, 1988.

[GMW91] O. Goldreich, S. Micali, and A. Wigderson. Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems. *Journal of the ACM*, 38:691–729, 1991.

[GNW95] O. Goldreich, N. Nisan, and A. Wigderson. On Yao's XOR-Lemma. Technical Report TR-95-050, Electronic Colloquium on Computational Complexity, 1995.

[GS89] S. Goldwasser and M. Sipser. Private coins versus public coins in interactive proof systems. In S. Micali, editor, *Randomness and Computation*, volume 5 of *Advances in Computing Research*, pages 73–90. JAI Press, Greenwich, 1989.

[Imp95] R. Impagliazzo. Hard-core distributions for somewhat hard problems. In *Proceedings of the 36th IEEE Symposium on Foundations of Computer Science*, pages 538–545, New York, 1995. IEEE.

[IW97] R. Impagliazzo and A. Wigderson. P=BPP unless E has sub-exponential circuits: Derandomizing the XOR lemma. In *Proceedings of the 29th ACM Symposium on the Theory of Computing*, pages 220–229, New York, 1997. ACM.

[Lev87] L. Levin. One-way functions and pseudo-random generators. *Combinatorica*, 7:357–363, 1987.

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

[NW94] N. Nisan and A. Wigderson. Hardness vs. randomness. *Journal of Computer and System Sciences*, 49:149–167, 1994.

[Pap94] Ch. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.

[Sha92] A. Shamir. IP = PSPACE. *Journal of the ACM*, 39:869–877, 1992.

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

Appendix: Proof of Lemma 3.1

Let L be a language satisfying (2) and (3), and consider

$$L' = \{\langle x, y\rangle \mid y \in \{0,1\}^{t(|x|)} \text{ and } \exists\, z \in \{0,1\}^{t(|x|)} : M(\langle x, y, z\rangle) = 1\}.$$

By Cook's Theorem [Coo88], since $L' \in \mathrm{NTIME}[n]$, there exists a circuit of size $t'(n) \in O(t(n)\log^2 t(n))$ that many-one reduces L' to SAT on inputs $\langle x, y\rangle$ with $x \in \{0,1\}^n$ and $y \in \{0,1\}^{t(n)}$. For any fixed value of $x \in \{0,1\}^n$, let C_x denote the corresponding oracle circuit obtained by hardwiring x. Note that C_x makes a single oracle query and outputs the answer to that query. Since G is a pseudo-random generator secure against parallel SAT, conditions (2) and (3) imply that

$$\Pr_{\sigma}[C_x^{\mathrm{SAT}}(G_{t'(n)}(\sigma)) = 1] \begin{array}{ccl} > & \frac{1}{2} & \text{if } x \in L \\ < & \frac{1}{2} & \text{if } x \notin L, \end{array} \tag{5}$$

where the probability is with respect to the uniform distribution of σ over $\{0,1\}^{s(t'(n))}$.

Since G is computable in $\mathrm{NTIME}[\tau(n)] \cap \mathrm{coNTIME}[\tau(n)]$, there exists a nondeterministic Turing machine N running in time τ that accepts $\{\langle n, \sigma, i, b\rangle \mid i\text{-th bit of } G_n(\sigma) \text{ equals } b\}$. Now, consider the nondeterministic algorithm in Figure 1.

Figure 1 describes a nondeterministic Turing machine that runs in time $2^{s(t'(n))} \cdot \tau(s(t'(n))) \cdot t(n)$. The largest possible value of counter at the end of the outer loop over all possible nondeterministic choices in the algorithm of Figure 1, equals $2^{s(t'(n))} \cdot \Pr_\sigma[C_x^{\mathrm{SAT}}(G_{t'(n)}(\sigma)) = 1]$. It follows from (5) that the machine accepts L. This finishes the proof of Lemma 3.1.

counter $\leftarrow 0$
for every $\sigma \in \{0,1\}^{s(t'(n))}$
  for $i \leftarrow 1, \ldots, t(n)$
    guess $\rho_i \in \{0,1\}$
    guess a computation path p for $N(\langle n, \sigma, i, \rho_i \rangle)$
    if $N(\langle n, \sigma, i, \rho_i \rangle)$ rejects along p
      then reject and abort
  guess $z \in \{0,1\}^{t(n)}$
  counter $\leftarrow$ counter $+ M(\langle x, \rho, z \rangle)$
if counter $> 2^{t'(n)-1}$ **then** accept

Figure 1: Nondeterministic algorithm for deciding L

Dieter van Melkebeek
Department of Computer Science
The University of Chicago
1100 East 58th Street
Chicago, IL 60637
USA
Email: dieter@cs.uchicago.edu
Web: http://www.cs.uchicago.edu/~dieter