THEORY OF COMPUTING, Volume 7 (2011), pp. 177–184
www.theoryofcomputing.org

NOTE

On Circuit Lower Bounds from Derandomization

Scott Aaronson* Dieter van Melkebeek†

Received: December 7, 2010; published: September 12, 2011.

Abstract: We present an alternate proof of the result by Kabanets and Impagliazzo (2004) that derandomizing polynomial identity testing implies circuit lower bounds. Our proof is simpler, scales better, and yields a somewhat stronger result than the original argument.

ACM Classification: F.1.2, F.1.3

AMS Classification: 68Q10, 68Q15, 68Q17

Key words and phrases: circuit lower bounds, derandomization, polynomial identity testing

1 Introduction

It is well-known that the standard approach for derandomizing BPP, namely via quick pseudorandom generators for circuits, requires proving superpolynomial circuit lower bounds for EXP (see [4], for example). For the larger class promise-BPP (in fact, for MA), Impagliazzo, Kabanets and Wigderson [3] showed that *any* derandomization—using pseudorandom generators or not—implies superpolynomial circuit lower bounds for NEXP. Building on that result, Kabanets and Impagliazzo [5] proved that *any* derandomization of BPP implies superpolynomial circuit lower bounds of some kind. More specifically, they showed that if polynomial identities over the integers can be decided deterministically (or even nondeterministically) in subexponential time, then either NEXP does not have Boolean circuits of polynomial size or the permanent over $\mathbb{Z}$ does not have arithmetic circuits of polynomial size. Note that

*Supported by NSF grant 0844626, a DARPA YFA grant, and the Keck Foundation.
†Supported by NSF grants 0728809 and 1017597.

 DOI: 10.4086/toc.2010.v007a012

BPP contains the language of all polynomial identities over the integers [2], so any derandomization of BPP implies a deterministic algorithm for deciding polynomial identities over the integers.

The proof in [5] hinges on the result from [3] that $\mathrm{NEXP} \subseteq \mathrm{P/poly} \Rightarrow \mathrm{NEXP} = \mathrm{MA}$, which itself involves the use of multi-prover interactive proofs for EXP (through the result that $\mathrm{EXP} \subseteq \mathrm{P/poly} \Rightarrow \mathrm{EXP} = \mathrm{MA}$ [1]), worst-case to average-case reductions for EXP (as in [1]), and the Nisan-Wigderson hardness-based pseudorandom generator construction [9, 10]. The proof in [5] also uses Toda's Theorem that the polynomial-time hierarchy reduces to counting [11]. The work on typically-correct derandomization by Kinne, Van Melkebeek, and Shaltiel [7] includes an alternate proof that avoids the most involved ingredient of the original proof, namely the result from [3], but still uses Toda's Theorem. In this note we present a yet simpler proof that avoids both [3] and Toda's Theorem. Our proof is completely elementary modulo the use of the #P-completeness of the permanent [12]. The only other nontrivial ingredients are that $\mathrm{NTIME}(2^{O(n)})$ has a complete problem under linear-time reductions, and Kannan's circuit lower bounds for Σ_2^{p} [6].

Apart from its simplicity and possible didactic value, our argument also yields a somewhat stronger result than the original one. It essentially gives the same lower bound for $\mathrm{NEXP} \cap \mathrm{coNEXP}$ as [5] does for NEXP. More importantly, our argument scales better than the original one, although the one in [7] scales even a bit better. We defer the discussion of these issues until after the presentation of the original and our new argument.

2 Notation

We adopt the following notation for the rest of this note. PIT stands for "polynomial identity testing" and is formalized as the language of all arithmetic circuits that compute the zero polynomial over $\mathbb{Z}$, where arithmetic circuits consist of internal nodes representing addition, subtraction, and multiplication, and leaves representing variables and the constants 0 and 1. PERM denotes the permanent of matrices over $\mathbb{Z}$. NSUBEXP is a shorthand for $\bigcap_{\varepsilon>0} \mathrm{NTIME}(2^{n^{\varepsilon}})$. For any function $s(n)$ we denote by $\mathrm{SIZE}(s(n))$ the class of languages L such that L at length n can be decided by a Boolean circuit of size $s(n)$ for all but finitely many input lengths n. We denote by $\mathrm{ASIZE}(a(n))$ the class of families $(p_n)_{n\in\mathbb{N}}$ of polynomials over $\mathbb{Z}$ where p_n has n variables and can be computed by an arithmetic circuit of size $a(n)$ for all but finitely many $n \in \mathbb{N}$.

Using the above notation we can state the result from [5] as follows.

Theorem 2.1 ([5])**.** *If* $\mathrm{PIT} \in \mathrm{NSUBEXP}$ *then*

(i) $\mathrm{NEXP} \not\subseteq \mathrm{SIZE}(\mathrm{poly}(n))$ *or*

(ii) $\mathrm{PERM} \notin \mathrm{ASIZE}(\mathrm{poly}(n))$.

3 The arguments

We now present the argument from [5] and our new argument for Theorem 2.1. The arguments have a similar structure and share a common part. They can both be cast as indirect diagonalization arguments—assume the theorem fails and obtain a contradiction with a diagonalization result. We break up the arguments into two steps, where the first step corresponds to the common part. In order to maximize the commonality, we present the argument from [5] slightly differently[1] than in that paper.

Step 1 Use the hypothesis that PERM has small arithmetic circuits, the checkability properties of PERM, and the hypothesis that PIT can be derandomized, to obtain an NSUBEXP algorithm for PERM and problems that efficiently reduce to PERM (such as all of Σ_2^{p}).

Step 2 Use the hypothesis that NEXP has small Boolean circuits and the algorithm from Step 1 to derive a contradiction with a diagonalization result.

Step 1 is common to both arguments. Step 2 in [5] uses the result from [3] that NEXP $\subseteq$ P/poly $\Rightarrow$ NEXP $=$ MA, whereas our alternate does not. We expand on Step 1 in Section 3.1, on Step 2 of [5] in Section 3.2, and on the new Step 2 in Section 3.3.

3.1 A common lemma

Lemma 3.1. *If* PIT $\in$ NSUBEXP *and* PERM $\in$ ASIZE(poly(n)), *then* $\Sigma_2^{\mathrm{p}} \subseteq$ NSUBEXP.

The proof of Lemma 3.1 relies on the following claim, which shows how checking the correctness of a purported arithmetic circuit for the permanent reduces to PIT.

Claim 3.2. *There exists a polynomial-time algorithm that takes an arithmetic circuit C and an integer m, and produces an arithmetic circuit $\tilde{C}$ such that C computes the permanent of $m \times m$ matrices over $\mathbb{Z}$ iff $\tilde{C} \in$ PIT.*

Claim 3.2 follows from viewing the Laplace expansion of the permanent as a downward self-reduction. We include a proof for completeness.

Proof of Claim 3.2. We use the following notation. Let M be an $m \times m$ matrix M. For $0 \le k \le m$ and $1 \le i, j \le k$, we let $M^{(k)}$ denote the matrix obtained by taking the $m \times m$ identity matrix and replacing the top left $k \times k$ submatrix by the corresponding submatrix of M. Let $M^{(k-1)}_{-i,-j}$ denote the same for $k-1$ but starting from the matrix M with the i-th row and j-th column deleted.

We have that C correctly computes the permanent of $m \times m$ matrices over $\mathbb{Z}$ iff for each $1 \le k \le m$, the polynomial

$$\tilde{C}_k = C(X^{(k)}) - \sum_{j=1}^{k} C(X^{(k-1)}_{-k,-j}) \cdot x_{kj}$$

is identically zero, as well as the polynomial $\tilde{C}_0 = C(X^{(0)}) - 1$, where X denotes an $m \times m$ matrix of variables $(x_{ij})_{i,j=1}^{m}$. By introducing one more variable x_0, those conditions can be expressed equivalently as whether the following polynomial is identically zero: $\tilde{C} = \sum_{k=0}^{m} \tilde{C}_k \cdot x_0^k$. The straightforward implementation of $\tilde{C}$ given C yields an arithmetic circuit that consists of $O(m^2)$ copies of C and some simple additional circuitry. That arithmetic circuit is in PIT iff C correctly computes the permanent on $m \times m$ matrices over $\mathbb{Z}$. □

[1] The difference is that [5] applies Toda's Theorem to simulate the polynomial-time hierarchy in $\mathrm{P}^{\#\mathrm{P}}$, and states and uses Lemma 3.1 for $\mathrm{P}^{\#\mathrm{P}}$ instead of Σ_2^{p}, but the application of Toda's Theorem can be avoided, and the rest of the argument can be adapted to work with Σ_2^{p} rather than $\mathrm{P}^{\#\mathrm{P}}$.

Given Claim 3.2, we prove Lemma 3.1 as follows.

Proof of Lemma 3.1. Assuming the hypotheses of the lemma, we will show that coNP $\subseteq$ NSUBEXP, which implies that $\Sigma_2^{\mathrm{p}} \subseteq$ NSUBEXP. The latter implication follows because for every $L \in \Sigma_2^{\mathrm{p}}$ there exists $L' \in$ coNP and a positive integer c such that for any input x:

$$\underbrace{x \in L \Leftrightarrow (\exists y \in \{0,1\}^{|x|^c}) \underbrace{\langle x,y \rangle \in L'}_{(*)}}_{(**)}. \tag{1}$$

Assuming that coNP $\subseteq$ NSUBEXP, we can replace the predicate $(*)$ in (1) by an NSUBEXP computation on the combined input $\langle x,y \rangle$, which turns $(**)$ into an NSUBEXP computation on input x.

For any language $L'' \in$ coNP there exists a function $f \in$ #P such that for any input x, $x \in L''$ iff $f(x) = 0$. Valiant's proof that the permanent is #P-complete [12] implies that for any $f \in$ #P there exists a polynomial-time computable mapping g onto square matrices with entries in $\{-1,0,1\}$ such that $f(x) = 0$ iff $\textsc{Perm}(g(x)) = 0$. Thus, it suffices to develop an NSUBEXP-algorithm to decide whether the permanent of a given $m \times m$ matrix M with entries in $\{-1,0,1\}$ is zero over the integers. We use the following procedure.

1. Guess a polynomial-sized candidate arithmetic circuit C for PERM on matrices of dimension m.
2. Verify the correctness of C. Halt and reject if the test fails.
3. Use the circuit C to determine the permanent of M and accept iff the result is 0.

The circuit in Step 1 exists by virtue of the hypothesis that PERM has polynomial-size arithmetic circuits. The combination of Claim 3.2 and the hypothesis that PIT $\in$ NSUBEXP shows how to do Step 2 in NSUBEXP. In order to execute Step 3 in deterministic polynomial time, we can evaluate the arithmetic circuit C on the given input M and perform the arithmetic modulo $m!+1$. The latter quantity exceeds $\textsc{Perm}(M)$ in absolute value, so the outcome of the computation is zero iff $\textsc{Perm}(M)$ is. Overall, the above 3-step procedure correctly decides whether $\textsc{Perm}(M) = 0$ in NSUBEXP. □

3.2 The original argument

Assume by way of contradiction that the hypothesis of Theorem 2.1 holds but that (i) and (ii) fail. We have that

$$\text{NEXP} \subseteq \text{MA} \subseteq \text{NSUBEXP}. \tag{2}$$

The first inclusion in (2) follows from [3] by our second hypothesis (NEXP $\subseteq$ SIZE(poly(n))). The second inclusion follows from Lemma 3.1 by our first hypothesis (PIT $\in$ NSUBEXP) and third hypothesis ($\textsc{Perm} \in$ ASIZE(poly(n))) and the fact that MA $\subseteq \Sigma_2^{\mathrm{p}}$. The two inclusions in (2) combined yield a contradiction to the nondeterministic time hierarchy theorem.[2]

[2]Alternately, since NEXP $\subseteq$ MA implies that NEXP = EXP and thus NEXP $\subseteq$ coMA, one can obtain the contradiction that NEXP $\subseteq$ coNSUBEXP, which is somewhat easier to refute (using "vanilla" diagonalization) than NEXP $\subseteq$ NSUBEXP, which involves "delayed" diagonalization.

3.3 The new argument

Assume by way of contradiction that the hypothesis of Theorem 2.1 holds but that (i) and (ii) fail. We have that

$$\Sigma_2^{\mathrm{p}} \subseteq \mathrm{NTIME}(2^{O(n)}) \subseteq \mathrm{SIZE}(n^c) \tag{3}$$

for some constant c. The first inclusion in (3) follows from Lemma 3.1 by our first hypothesis (PIT $\in$ NSUBEXP) and our third hypothesis (PERM $\in$ ASIZE(poly(n))). The second inclusion follows from our second hypothesis (NEXP $\subseteq$ SIZE(poly(n))) and the fact that NTIME($2^{O(n)}$) has a complete problem under linear-time reductions,[3] e. g.,

$$\{\langle M,x,t\rangle \mid M \text{ is a nondeterministic Turing machine that accepts } x \text{ in at most } t \text{ steps}\}.$$

All together (3) yields a contradiction to the result of Kannan's that for every fixed constant c there exists a language in Σ_2^{p} that does not have Boolean circuits of size n^c [6].

4 Strengthening and parameterized statement

Kannan actually showed that $\Sigma_2^{\mathrm{p}} \cap \Pi_2^{\mathrm{p}}$ (rather than Σ_2^{p}) is not in SIZE(n^c) for any constant c. Incorporating that fact into our argument leads to a strengthening that essentially[4] allows us to replace NEXP in part (i) of Theorem 2.1 by NEXP$\cap$coNEXP.

Both the original and the new argument can trade the running time of the nondeterministic algorithm for PIT and the size of the arithmetic circuits for PERM in part (ii). However, due to the use of the implication NEXP $\subseteq$ SIZE(poly(n)) $\Rightarrow$ NEXP $=$ MA from [3], the original argument does not accommodate changes to either the right-hand side or the left-hand side of (i), whereas the new argument allows us to play with both sides. On the left-hand side, the proof in [3] can only handle time bounds that are at least exponential.[5] This is true even when the running time of the nondeterministic algorithm for PIT is polynomial, in which case our argument only needs the time bound on the left-hand side of (i) to be superpolynomial. On the right-hand side, the proof in [3] can only handle circuit sizes that are polynomial;[6] our proof gives nontrivial results for circuit sizes ranging from linear to linear-exponential.

We can further improve the parameterized version of Theorem 2.1 by slightly modifying our argument and incorporating Toda's Theorem [11]. The strengthening and improved parameterization are captured in the following statement, which appears in [7]. We use (N$\cap$coN)TIME(τ) as a shorthand for NTIME(τ)$\cap$ coNTIME(τ).

[3]To complete the comparison, we point out that the argument for the second inclusion is also used as a step in the proof in [3] that NEXP $\subseteq$ P/poly $\Rightarrow$ NEXP $=$ MA.

[4]More precisely, this modification allows us to replace NEXP by NEXP$\cap$coNEXP under the slightly stronger hypothesis that PIT $\in$ NTIME($t(n)$) for some constructible $t(n)$ that is $O(2^{n^\varepsilon})$ for every positive ε. Issues of uniformity make the argument somewhat tedious. For that reason and the fact that modification using Toda's Theorem (discussed next) avoids those issues as well as the need for the slightly stronger hypothesis, we do not spell out the proof.

[5]This is because of the use of the implication EXP $\subseteq$ SIZE(poly(n)) $\Rightarrow$ EXP $=$ MA, in which we do not know whether we can replace EXP by $\mathcal{C} = \mathrm{DTIME}(t(n))$ for subexponential $t(n)$, as the argument needs multiple-prover interactive proofs for $\mathcal{C}$ with honest provers computable in $\mathrm{P}^{\mathcal{C}}$.

[6]This is because the argument in [3] first shows that the hypothesis implies that NEXP $=$ EXP and then resorts to the implication EXP $\subseteq$ SIZE(poly(n)) $\Rightarrow$ EXP $=$ MA. The first step involves cycling over all circuits of the size given by the right-hand side of the premise. This can be done in EXP only when that size is polynomial.

Theorem 4.1 (Theorem 9 in [7]). *Let $\gamma(n)$ denote the maximum circuit complexity of Boolean functions on n inputs. There exists a constant $c > 0$ such that the following holds for any functions $a(n)$, $s(n)$, and $t(n)$ such that $a(n)$ and $s(n)$ are constructible, $a(n)$ and $t(n)$ are monotone, and $n \leq s(n) < \gamma(n)$.*

If $\mathrm{PIT} \in \mathrm{NTIME}(t(n))$ *then*

(i) $(\mathrm{N} \cap \mathrm{coN})\mathrm{TIME}(t((s(n))^c \cdot a((s(n))^c))) \not\subseteq \mathrm{SIZE}(s(n))$, *or*

(ii) $\textsc{Perm} \notin \mathrm{ASIZE}(a(n))$.

The instantiation of Theorem 4.1 to the setting of Theorem 2.1 yields the following statement.

Corollary 4.2. *If* $\mathrm{PIT} \in \mathrm{NSUBEXP}$ *then*

(i) $\mathrm{NEXP} \cap \mathrm{coNEXP} \not\subseteq \mathrm{SIZE}(\mathrm{poly}(n))$ *or*

(ii) $\textsc{Perm} \notin \mathrm{ASIZE}(\mathrm{poly}(n))$.

We refer to [7] for the detailed analysis and formal proof of Theorem 4.1. Here we merely present a sketch of the modifications to our proof of Theorem 2.1 and an explanation of how Toda's Theorem helps. We point out that the focus in [7] lies on so-called typically-correct derandomization. In particular, [7] shows that Theorem 2.1 holds even under the weaker hypothesis that for every positive ε there exists a nondeterministic algorithm that runs in time 2^{n^ε} and decides PIT correctly on all but 2^{n^ε} of the inputs of length n for all but finitely many n.

The weakness of the argument without Toda's Theorem stems from the use of Kannan's result that $\Sigma_2^{\mathrm{p}} \not\subseteq \mathrm{SIZE}(n^c)$ for any constant c, a result that does not scale as well as one might hope (see [8] for a discussion). The issue disappears when we go higher up in the polynomial-time hierarchy, where Kannan's argument generalizes to $\Sigma_3^{\mathrm{p}}\mathrm{TIME}((s(n))^2 \log^a s(n)) \not\subseteq \mathrm{SIZE}(s(n))$ and $\Sigma_4^{\mathrm{p}}\mathrm{TIME}(s(n) \log^a s(n)) \not\subseteq \mathrm{SIZE}(s(n))$ for some constant a and any constructible bound $s(n)$ less than the maximum circuit complexity. By Toda's Theorem there exists a constant b and a problem $A \in \#\mathrm{P}$ such that $\Sigma_4^{\mathrm{p}}\mathrm{TIME}(t(n)) \subseteq \mathrm{DTIME}^A((t(n))^b)$ for any constructible bound $t(n)$. The proof of Lemma 3.1 shows that its statement holds even when we replace Σ_2^{p} by $\mathrm{P}^{\#\mathrm{P}}$, and the argument scales optimally. The fact that $\mathrm{P}^{\#\mathrm{P}}$ is closed under complementation automatically leads to a simulation in $(\mathrm{N} \cap \mathrm{coN})\mathrm{TIME}(\cdot)$ in the generalization of Lemma 3.1. Combining all the ingredients in a similar way as in our proof of Theorem 2.1 yields the statement of Theorem 4.1.

Acknowledgements

We would like to thank Valentine Kabanets for encouraging us to write this note, and the anonymous referees for many helpful suggestions on the exposition. DvM is grateful to Jeff Kinne and Ronen Shaltiel for helpful discussions and comments and for the related joint work in [7].

References

[1] LÁSZLÓ BABAI, LANCE FORTNOW, NOAM NISAN, AND AVI WIGDERSON: BPP has subexponential time simulations unless EXPTIME has publishable proofs. *Comput. Complexity*, 3:307–318, 1993. 178

[2] OSCAR IBARRA AND SHLOMO MORAN: Probabilistic algorithms for deciding equivalence of straight-line programs. *J. ACM*, 30(1):217–228, 1983. [doi:10.1145/322358.322373] 178

[3] RUSSELL IMPAGLIAZZO, VALENTINE KABANETS, AND AVI WIGDERSON: In search of an easy witness: Exponential time vs. probabilistic polynomial time. *J. Comput. System Sci.*, 65(4):672–694, 2002. [doi:10.1016/S0022-0000(02)00024-7] 177, 178, 179, 180, 181

[4] RUSSELL IMPAGLIAZZO, RONEN SHALTIEL, AND AVI WIGDERSON: Reducing the seed length in the Nisan-Wigderson generators. *Combinatorica*, 26(2):647–681, 2006. [doi:10.1016/S0022-0000(02)00024-7] 177

[5] VALENTINE KABANETS AND RUSSELL IMPAGLIAZZO: Derandomizing polynomial identity tests means proving circuit lower bounds. *Comput. Complexity*, 13(1/2):1–46, 2004. [doi:10.1007/s00037-004-0182-6] 177, 178, 179

[6] RAVI KANNAN: Circuit-size lower bounds and nonreducibility to sparse sets. *Inf. Control*, 55(1–3):40–56, 1982. [doi:10.1016/S0019-9958(82)90382-5] 178, 181

[7] JEFF KINNE, DIETER VAN MELKEBEEK, AND RONEN SHALTIEL: Pseudorandom generators, typically-correct derandomization, and circuit lower bounds. Technical Report TR10-129, Electron. Colloq. on Comput. Complexity (ECCC), 2010. A preliminary version of this paper appeared in the proceedings of *RANDOM '09*; the full version will appear in a special issue of *Computational Complexity*. [ECCC:TR10-129] 178, 181, 182

[8] PETER BRO MILTERSEN, N. V. VINODCHANDRAN, AND OSAMU WATANABE: Super-polynomial versus half-exponential circuit size in the exponential hierarchy. In *Proc. 5th Ann. Intern. Conf. Computing and Combinatorics (COCOON '99)*, number 1627 in Lecture Notes in Comput. Sci., pp. 210–220. Springer, 1999. [doi:10.1007/3-540-48686-0_21] 182

[9] NOAM NISAN: Pseudorandom bits for constant depth circuits. *Combinatorica*, 11(1):63–70, 1991. [doi:10.1016/S0022-0000(02)00024-7] 178

[10] NOAM NISAN AND AVI WIGDERSON: Hardness vs. randomness. *J. Comput. System Sci.*, 49(2):149–167, 1994. [doi:10.1016/S0022-0000(05)80043-1] 178

[11] SEINOSUKE TODA: PP is as hard as the polynomial-time hierarchy. *SIAM J. Comput.*, 20(5):865–877, 1991. [doi:10.1137/0220053] 178, 181

[12] LESLIE G. VALIANT: The complexity of computing the permanent. *Theoret. Comput. Sci.*, 8(2):189–201, 1979. [doi:10.1016/S0022-0000(02)00024-7] 178, 180

AUTHORS

Scott Aaronson
associate professor
Department of Electrical Engineering and Computer Science
Massachusetts Institute of Technology
Cambridge, Massachusetts, USA
aaronson@csail.mit.edu
http://www.scottaaronson.com

Dieter van Melkebeek
associate professor
Department of Computer Sciences
University of Wisconsin-Madison
Madison, Wisconsin, USA
dieter@cs.wisc.edu
http://pages.cs.wisc.edu/~dieter

ABOUT THE AUTHORS

SCOTT AARONSON received his Ph. D. from the University of California at Berkeley, under the supervision of Umesh Vazirani. After postdocs at the University of Waterloo and the Institute for Advanced Study, he joined the faculty at MIT. He writes the blog Shtetl-Optimized.

DIETER VAN MELKEBEEK received his Ph. D. from the University of Chicago, under the supervision of Lance Fortnow. His thesis was awarded the ACM Doctoral Dissertation Award. After postdocs at DIMACS and the Institute for Advanced Study, he joined the faculty at the University of Wisconsin-Madison.