Electronic Colloquium on Computational Complexity, Report No. 157 (2006)

Fixed-Polynomial Size Circuit Bounds

Lance Fortnow
University of Chicago
fortnow@cs.uchicago.edu

Rahul Santhanam
Simon Fraser University
rsanthan@cs.sfu.ca

Abstract

We explore whether various complexity classes can have linear or more generally n^k-sized circuit families for some fixed k. We show

- The following are equivalent,
  - NP is in SIZE(n^k) (has $O(n^k)$-size circuit families) for some k
  - $\mathsf{P}_{\|}^{\mathsf{NP}}$ is in SIZE(n^k) for some k
  - ONP/1 is in SIZE(n^k) for some k.

  where ONP is the class of languages accepted by NP machines with some witness depending only on the input length.
- For all k, MA is in SIZE(n^k) if and only if AM is in SIZE(n^k).
- One cannot show $\oplus$P does not have n^2-size circuit families without using nonrelativizing techniques beyond those already used for similar results.
- For every k, the class P^{PP} does not have n^k-sized circuits with $\Sigma_k^{\oplus\mathsf{P}}$-gates.
- For a large number of natural classes $\mathcal{C}$ and constant k, $\mathcal{C}$ is in SIZE(n^k) if and only if $\mathcal{C}/1 \cap \mathsf{P/poly}$ is in SIZE(n^k).

1 Introduction

Proving lower bounds for general nonuniform circuit results remains one of the most difficult tasks in computational complexity. One has to go to the exponential-time version of Merlin-Arthur games to find a class provably not having a polynomial-size circuit family [BFT98]. We currently do not have any techniques for showing EXP cannot have poly-size circuits and certainly not super-polynomial lower bounds for NP needed to settle the P versus NP question.

What if instead we ask to show that a class doesn't have n^k-sized circuits for some fixed k, like whether NP has a quadratic-sized circuit family. That question remains open but we have seen some progress on larger classes. In 1982, Kannan [Kan82] showed that Σ_2^p does not have n^k-sized circuits for any k. In 2005, Vinodchandran [Vin05] showed that the class PP does not have n^k-sized circuits. Very recently Santhanam [San06] improved both results by showing that the promise version of MA does not have n^k circuits for any fixed k.

Can we prove a similar result for other classes like NP, $\mathsf{P}_{\|}^{\mathsf{NP}}$, $\oplus$P, AM and MA? These are the questions we explore in this paper. We don't settle any of these problems but we show several connections between them and try to capture the limitations of known techniques.

Showing the separation for $\mathsf{P}_{\|}^{\mathsf{NP}}$ would imply the separation for NP, more specifically, if $\mathsf{P}_{\|}^{\mathsf{NP}}$ does not have a n^k-size circuit family for any constant k then neither does NP. Similarly we show that if AM does not have n^k-size circuits then neither does MA.

ISSN 1433-8092

We explore the class ONP, "Oblivious" NP, implicitly defined by Chakaravarthy and Roy [CR06]. A language L is in ONP if for every n there is a single witness w that witnesses every x in L with $|x| = n$. We show that ONP nearly captures the hardness of showing NP does not have small circuits: If NP does not have n^k-sized circuits then ONP/1 does not have n^k-sized circuits.

Vinodchandran and Santhanam's lower bounds [Vin05, San06] use nonrelativizing techniques which suggests that oracle results cannot tell us much about the limitations of separation theorems. The only nonrelativizing techniques they use is based on interactive proof systems (see [BFL91b]), arguably the only true nonrelativizing technique currently available in computational complexity. We exhibit a relativized world where $\oplus$P has a n^2-sized circuit family and the conclusions of the nonrelativizing techniques used by Vinodchandran and Santhanam also hold. This means that to prove $\oplus$P does not have quadratic-size circuit families would require nonrelativizing techniques beyond those already known.

We give lower bounds against stronger circuit models. We show that the class P^{PP} does not have small circuits with $\oplus$P or Σ_k^p gates and more generally for any k, P^{PP} does not have n^k-size circuits with $\Sigma_k^{\oplus\mathsf{P}}$-gates.

Finally we prove a general result for a wide variety of complexity classes such as NP, P^{NP}, MA, BPP and PP. For all these classes $\mathcal{C}$ and many more, if $\mathcal{C}$ does not have n^k-size circuit families then $\mathcal{C}/1 \cap \mathsf{P}/\mathsf{poly}$ does not have n^k-size circuits either.

2 Preliminaries

2.1 Complexity Classes, Promise Problems and Advice

We assume a basic familiarity with complexity classes such as P, RP, BPP, NP, MA, AM, Σ_2,PP, $\sharp P$ and PSPACE. The Complexity Zoo (`http://qwiki.caltech.edu/wiki/ComplexityZoo`) is an excellent resource for basic definitions and statements of results.

We also use simultaneous time and space bounded classes. $\Sigma_k\mathsf{TISP}(T, S)$ is the class of languages accepted by Σ_k machines operating simultaneously in time T and space S.

Given a complexity class C, coC is the class of languages L such that $\bar{L} \in \mathsf{C}$. Given a function $s : \mathbb{N} \to \mathbb{N}$, SIZE($s$) is the class of Boolean functions $f = \{f_n\}$ such that for each n, f_n has Boolean circuits of size $O(s(n))$. For a Boolean function f, $Ckt(f)$ is the circuit complexity of f, i.e., the size of the smallest circuit computing f, Given a language L and an integer n, $L_n = L \cap \{0,1\}^n$.

We also require the notion of circuit size for other circuit models. These are typically defined by having one or more auxiliary inputs to a deterministic circuit and defining the language accepted by the circuit using some condition on acceptance of auxiliary inputs. $\oplus$SIZE(s) is the class of Boolean functions computed by Parity-circuits of size $O(s)$, i.e., an input x is accepted by the circuit if the circuit accepts on an odd number of auxiliary inputs. Similarly, we define the classes $\Sigma_a - \mathsf{SIZE}(s)$ for any integer a, and PSIZE(s), where the circuit accepts an input if a majority of auxiliary inputs are accepted.

In order to deal with promise classes in a general way, we take as fundamental the notion of a complexity measure. A complexity measure CTIME is a mapping which assigns to each pair (M, x), where M is a time-bounded machine (here a time function $t_M(x)$ is implicit) and x an input, one of three values "0" (accept), "1" (reject) and "?" (failure of CTIME promise). We distinguish between *syntactic* and *semantic* complexity measures. Syntactic measures have as their range $\{0, 1\}$ while semantic measures may map some machine-input pairs to "?". The complexity measures DTIME and NTIME are syntactic (each halting deterministic or non-deterministic machine either accepts or rejects on each input), while complexity measures such as BPTIME and MATIME are semantic

(a probabilistic machine may accept on an input with probability 1/2, thus failing the bounded-error promise). For syntactic measures, any halting machine defines a language, while for semantic measures, only a subset of halting machines define languages.

A promise problem is a pair (Y, N), where $Y, N \subseteq \{0,1\}^*$ and $Y \cap N = \varnothing$. We say that a promise problem (Y, N) belongs to a class $\mathsf{CTIME}(t)$ if there is a machine M halting in time t on all inputs of length n such that M fulfils the CTIME promise on inputs in $Y \cup N$, accepting on inputs in Y and rejecting on inputs in N.

For a complexity class C, $\mathsf{Promise}-\mathsf{C}$ is the class of promise problems which belong to C. Sometimes, when C is a syntactic class, we abuse notation and use C and $\mathsf{Promise}-\mathsf{C}$ interchangeably.

A language L is in $\mathsf{CTIME}(t)/a$ if there is a machine M halting in time t taking an auxiliary *advice* string of length a such that for each n, there is some advice string $b_n, |b_n| = a$ such that M fulfils the CTIME promise for each input x with advice string b_n and accepts x iff $x \in L$.

For syntactic classes, a lower bound with advice or for the promise version of the class translates to a lower bound for the class itself.

Proposition 1. *Let* CTIME *be a syntactic complexity measure. If* $\mathsf{CTIME}(\mathsf{poly}(n))/O(n) \not\subseteq \mathsf{SIZE}(s(n))$, *then* $\mathsf{CTIME}(\mathsf{poly}(n)) \not\subseteq \mathsf{SIZE}(s(o(n)))$.

Proposition 2. *Let* CTIME *be a syntactic complexity measure. If* $\mathsf{Promise}-\mathsf{CTIME}(\mathsf{poly}(n)) \not\subseteq \mathsf{SIZE}(s(n))$, *then* $\mathsf{CTIME}(\mathsf{poly}(n)) \not\subseteq \mathsf{SIZE}(s(n))$.

2.2 Oblivious Classes

Intuitively, if a class C is defined using "proofs of acceptance" for each input and some condition on the verifiability of proofs, the oblivious version of the class C is the class of languages for which the *same* proof can be used on any input of a certain length. Oblivious versions of symmetric alternation classes were defined by Chakaravarthy and Roy [CR06] for the purpose of obtaining tight uniform characterizations of $\mathsf{NP} \subseteq \mathsf{SIZE}(\mathsf{poly})$. Here, we extend the definitions to non-deterministic classes (punctually!), and to Merlin-Arthur classes.

Definition 3. *We say a language L is in* $\mathsf{ONTIME}(t)$ *if there is a relation $R(x, y)$ computable in deterministic time $t(|x|)$, and a sequence of witnesses $\{w_n\}, n = 1 \ldots \infty$, such that $x \in L$ iff $R(x, w_{|x|})$ holds.*

Definition 4. *We say a language L is in* $\mathsf{OMATIME}(t)$ *if there is a relation $R(x, y, z)$ computable in deterministic time $t(|x|)$ and a sequence of witnesses $\{w_n\}, n = 1 \ldots \infty$, such that:*

1. *If $x \in L$, then for all z, $R(x, w_{|x|}, z)$ holds.*
2. *If $x \notin L$, then for any y, $\Pr_z R(x, y, z) < 1/2$.*

We have that $\mathsf{ONP} \subseteq \mathsf{OMA} \subseteq \mathsf{SIZE}(\mathsf{poly})$. The first inclusion is immediate; for the second inclusion, note that we can amplify the success probability of an OMA protocol above $1 - 2^{-n}$ just as we do for an MA protocol. By the union bound, there must be some random string z that gives the correct answer for every input when we have guessed the oblivious witness y. Giving y and z as advice for each input length is sufficient to decide the language.

On the other hand, all sparse languages in NP are contained in ONP, and all sparse languages in MA are contained in OMA. Thus we do not expect either of these classes to be easy - indeed, $\mathsf{ONP} = \mathsf{P}$ implies $\mathsf{NEXP} = \mathsf{EXP}$. Nor is it likely to be easy to show $\mathsf{OMA} \subseteq \mathsf{NP}$, since that would imply $\mathsf{MAE} = \mathsf{NE}$, and solve long-standing derandomization questions.

Using the notions above, we can get tight uniform characterizations of $\mathsf{C} \subseteq \mathsf{SIZE}(\mathsf{poly})$ for several interesting classes C.

Proposition 5. $\mathsf{NP} \subseteq \mathsf{SIZE}(\mathsf{poly})$ *iff* $\mathsf{NP} \subseteq \mathsf{ONP}$ *iff* $\mathsf{NP} \subseteq \mathsf{OMA}$.

Proof. From the preceding discussion, it is clear that $\mathsf{NP} \subseteq \mathsf{ONP}$ implies $\mathsf{NP} \subseteq \mathsf{OMA}$, and $\mathsf{NP} \subseteq \mathsf{OMA}$ implies $\mathsf{NP} \subseteq \mathsf{SIZE}(\mathsf{poly})$. Thus we just need to show that $\mathsf{NP} \subseteq \mathsf{SIZE}(\mathsf{poly})$ implies $\mathsf{NP} = \mathsf{ONP}$. We will show that under this assumption, SAT $\in \mathsf{ONP}$, and then use the fact that ONP is closed under m-reductions to conclude $\mathsf{NP} = \mathsf{ONP}$.

Assume SAT $\in \mathsf{SIZE}(n^k)$ for some k. We define the following ONP machine M for SAT. Given a formula ϕ of size n, M guesses a circuit C of size n^k for SAT on inputs of length n. If C accepts on ϕ, M uses C to find a candidate satisfying assignment w via self-reducibility and paddability of SAT. If w is a valid assignment, M accepts, otherwise it rejects. Note that no unsatisfiable formula is ever accepted in this process, moreover if C is a correct circuit for SAT, all satisfiable formulae are accepted. Thus C is an oblivious witness for SAT on length n. □

Proposition 6. $\mathsf{EXP} \subseteq \mathsf{SIZE}(\mathsf{poly})$ *iff* $\mathsf{EXP} = \mathsf{OMA}$.

Proof. The backward direction follows since $\mathsf{OMA} \subseteq \mathsf{SIZE}(\mathsf{poly})$. We show the forward direction. It follows from work on instance checkers and interactive proofs [BFL91b, BFNW93] that if $\mathsf{EXP} \subseteq \mathsf{SIZE}(\mathsf{poly})$, then $\mathsf{EXP} = \mathsf{MA}$. The proof of this result also gives $\mathsf{EXP} = \mathsf{OMA}$. □

Since Impagliazzo, Kabanets and Wigderson [IKW02] showed that $\mathsf{NEXP} \subseteq \mathsf{SIZE}(\mathsf{poly})$ implies $\mathsf{NEXP} = \mathsf{EXP}$ we have the following corollary.

Corollary 7. $\mathsf{NEXP} \subseteq \mathsf{SIZE}(\mathsf{poly})$ *iff* $\mathsf{NEXP} = \mathsf{MA}$ *iff* $\mathsf{NEXP} = \mathsf{OMA}$.

3 Collapses of Circuit Lower Bounds

The question of fixed polynomial size circuit lower bounds was first considered by Kanan, who proved lower bounds for Σ_2.

Theorem 8. *[Kan82] For any $k > 0$, $\Sigma_2 \not\subseteq \mathsf{SIZE}(n^k)$.*

Theorem 8 has been strengthened progressively in a sequence of papers [BCG+96, KW98, Cai01], and the smallest uniform complexity class for which we can show unconditional lower bounds is $\mathsf{S}_2^{\mathsf{P}}$ [Cai01]. Circuit lower bounds have recently been shown for the promise version of MA [San06] but showing such lower bounds for uniform MA and smaller classes remains an important open question. Such lower bounds for NP, apart from being interesting in their own right, would also separate BPP and NEXP, which would be a major breakthrough in the area of derandomization.

One obstacle to proving lower bounds for classes smaller than $\mathsf{S}_2^{\mathsf{P}}$ is that such results cannot relativize. There have been non-relativizing results in this area [Vin05, San06], but there is a paucity of non-relativizing techniques apart from the arithmetization technique used in work on interactive proofs [LFKN92, Sha92].

Given the difficulty of actually proving circuit lower bounds, we settle for reductions between them. We show for various pairs of classes B and C, where $\mathsf{B} \subseteq \mathsf{C}$ that a fixed polynomial lower bound for C also implies a fixed polynomial lower bound for the smaller class B. We call such results *collapses* of circuit lower bounds.

One example of a collapse is the result that if the polynomial hierarchy contains a language of superpolynomial circuit complexity, then so does NP. However, this result no longer holds if we consider fixed polynomial size. Indeed, if it did, we would already have a superlinear circuit lower bound for NP, by Theorem 8.

Our first result collapses circuit lower bounds for AM to circuit lower bounds for MA.

Theorem 9. *For any $k > 0$, $\mathsf{AM} \not\subseteq \mathsf{SIZE}(n^k)$ iff $\mathsf{MA} \not\subseteq \mathsf{SIZE}(n^k)$.*

Proof. The forward direction follows from the fact that $\mathsf{MA} \subseteq \mathsf{AM}$ [Bab85]. For the other direction if $MA \subseteq \mathsf{SIZE}(n^k)$ then $\mathsf{NP} \subseteq \mathsf{SIZE}(\mathsf{poly})$ which implies $\mathsf{AM} = \mathsf{MA}$ [AKSS95]. □

Next, we consider fixed polynomial size lower bounds for the class $\mathsf{P}_{\|}^{\mathsf{NP}}$, which lies between NP and P^{NP}. We show that such lower bounds would in fact imply lower bounds for NP.

Theorem 10. *If $\mathsf{P}_{\|}^{\mathsf{NP}} \not\subseteq \mathsf{SIZE}(n^k)$ for any $k > 0$, then $\mathsf{NP} \not\subseteq \mathsf{SIZE}(n^k)$ for any $k > 0$.*

Proof. We show that if NP has circuits of size n^k for some $k > 0$, then $\mathsf{P}_{\|}^{\mathsf{NP}}$ has circuits of size $n^k\mathsf{polylog}(n)$. This implies the theorem.

Let $L \in \mathsf{P}_{\|}^{\mathsf{NP}}$, and M be a polynomial-time machine deciding L with parallel access to SAT. We construct a circuit of size $n^k\mathsf{polylog}(n)$ deciding L. In order to do this, we first define a quasilinear-time oracle machine M' and two NP languages L_1 and L_2 such that M' decides L when given oracle access to L_1 and L_2.

Let x be an input of length n and $q_1 \ldots q_k$ be the oracle queries of M on input x, where $k = O(\mathsf{poly}(n))$. We define the L_1 as the set of pairs $\langle x, r\rangle$ such that at least r queries that $M(x)$ makes to SAT are satisfiable.

Clearly, $L_1 \in \mathsf{NP}$ since a non-deterministic Turing machine can generate the queries q_i and guess satisfying assignments for r of them. If at least r of the q_i's are satisfiable, then there is a sequence of guesses that leads to acceptance, otherwise there isn't.

We define L_2 using a non-deterministic Turing machine M_2. Given input $\langle x, r\rangle$, M_2 generates queries $q_1 \ldots q_k$, guesses satisfying assignments for exactly r of the q_i's, and simulates M assuming all other queries are answered 0. It accepts if M accepts. It rejects if it fails to guess satisfying assignments for r different q_i's, or if M rejects at the end of the simulation. By definition, $L_2 \in \mathsf{NP}$.

Now we define the behavior of M'. Given an input x, M' finds by binary search the greatest r such that $\langle x, r\rangle \in L_1$, using oracle queries to L_1. Since $k = O(\mathsf{poly}(n))$, the binary search can be done in $O(n \log(n))$ time using $O(\log(n))$ queries to L_1. Having found r, M' determines if $\langle x, r\rangle \in L_2$, using one oracle query to L_2. If the oracle query returns "yes", M' accepts, otherwise it rejects.

To see that M' decides correctly that $x \in L$, note that using the binary search, it obtains the correct number r of queries q_i that are in SAT. Hence, when M_2 guesses r satisfying assignments for queries $q_{i_1} \ldots q_{i_r}$, these must be *precisely* the queries that are satisfiable, and the other queries must be unsatisfiable. Hence, given $\langle x, r\rangle$, where r is the correct number of satisfiable queries, M_2 accepts iff M accepts x.

Now, by assumption, there are circuits C_1 and C_2 of size $O(n^k)$ for L_1 and L_2 respectively. We convert the oracle machine M' into an oracle circuit (with at most a polylogarithmic overhead in size), and replace oracle queries in the circuit with copies of C_1 or C_2, as the case may be. Since M' makes $O(\log(n))$ calls to L_1 and 1 call to L_2, the size of the resulting circuit is at most $n^k\mathsf{polylog}(n)$. □

The next result show that fixed polynomial lower bounds for NP also collapse, to fixed polynomial lower bounds for the oblivious version of NP (using 1 bit of advice). An interesting aspect of this result is that it illustrates that proving superpolynomial circuit lower bounds is a very different problem than proving fixed polynomial lower bounds. On the one hand, $\mathsf{ONP}/1 \subseteq \mathsf{SIZE}(\mathsf{poly})$ and we do not expect $\mathsf{NP} \subseteq \mathsf{SIZE}(\mathsf{poly})$, thus the two inclusions are very unlikely to be equivalent. On the other hand, the inclusions of the two classes in fixed polynomial size are equivalent.

Theorem 11. *For any k,* $\mathsf{NP} \not\subseteq \mathsf{SIZE}(n^k)$ *if and only if* $\mathsf{ONP}/1 \not\subseteq \mathsf{SIZE}(n^k)$.

Proof. The "if" direction is easy. If $\mathsf{ONP}/1$ does not have circuits of size n^k, then it follows that $\mathsf{NP}/1$ does not have circuits of size n^k. Since NP is a syntactic class, this implies that NP does not have circuits of size n^k.

The other direction is more involved. Assume NP does not have circuits of size n^k. We consider two cases. If $\mathsf{NP} \subseteq \mathsf{SIZE}(\mathsf{poly})$, then by Proposition 5 $\mathsf{NP} = \mathsf{ONP}$, and hence ONP does not have circuits of size n^k.

If $\mathsf{NP} \not\subseteq \mathsf{SIZE}(\mathsf{poly})$, then SAT $\notin \mathsf{SIZE}(\mathsf{poly})$. We use the fact that there is a "smoothly parametrized" version of Proposition 5. If $\mathsf{NP} \subseteq \mathsf{SIZE}(s)$, then we get $\mathsf{NP} \subseteq \mathsf{ONTIME}(\mathsf{poly}(s))$, for *arbitrary* circuit size s. By letting s be the circuit complexity of SAT, we get that SAT $\in \mathsf{ONTIME}(\mathsf{poly}(s))$ but SAT does not have circuits of size $s-1$. We then scale this separation down using an advice-efficient padding argument to conclude that a padded version of SAT is in ONP but does not have circuits of size $O(n^k)$.

The above is a brief sketch. We now proceed more formally. We define the following language L:

$$L = \{x1^r | x \in \text{SAT}, r \text{ is a power of } 2, r \geqslant |x|, Ckt(\text{SAT}_{|x|}) \leqslant (|x| + r)^{2k}\}$$

First we show $L \in \mathsf{ONP}/1$, and then we show $L \notin \mathsf{SIZE}(n^k)$.

We define a non-deterministic polynomial time machine M taking one bit of advice, such that when the advice bit is correct for length n, there is a polynomial-size witness w_n which works for any input of that length. Given an input y of length n, M first checks if it can be decomposed as $x1^r$ for r a power of 2, such that $r \geqslant |x|$. For any input y, there can be at most one such decomposition. This check can be performed in linear time, and if it succeeds, the corresponding x and r can be obtained. M uses its bit of advice to check if $Ckt(\text{SAT}_{|x|}) \leqslant (|x| + r)^{2k}$. This is just one bit of information given n since n uniquely determines $|x|$ and r. If the check using the advice fails, M rejects. Otherwise, M guesses a circuit C of size n^{2k}. It simulates C on x. If C accepts on x, it uses self-reducibility and paddability of SAT to find a candidate satisfying assignment for x. If the assignment works, M accepts, otherwise M rejects.

Clearly, M runs in polynomial time. Also, there is a single witness of size $\mathsf{poly}(n)$, namely a correct circuit C for SATon inputs of length $|x|$ which works for any input $x1^r \in L$, when M is given the correct bit of advice. Thus $L \in \mathsf{ONP}/1$.

Assume, for the purpose of contradiction, that $L \in \mathsf{SIZE}(n^k)$. Hence there is a sequence of circuits D_n of size $O(n^k)$ deciding L_n for each n. We show that this implies that there for infinitely many m, there is a circuit C_m of size less than $Ckt(\text{SAT}_m)$ deciding SATon inputs of length m. We define the circuits C_n as follows. Given an input x of length m, we non-uniformly determine the least $r(m)$ a power of 2 such that $r(m) \geqslant m$ and $Ckt(\text{SAT}_m) \leqslant (m + r(m))^{2k}$. Such an $r(m)$ exists for each m. Also, there are infinitely many m such that $r(m) > 2m$, for otherwise $Ckt(\text{SAT}_m) \leqslant (3m)^{2k} = O(\mathsf{poly}(m))$, which is a contradiction to our assumption that SAT does not have polynomial-size circuits. Now, for each m such that $r(m) > 2m$, it must be the case that $Ckt(\text{SAT}_m) > (m + r(m)/2)^{2k}$, just by assumption on minimality of $r(m)$. Thus, for these m, it must be the case that $Ckt(\text{SAT}_m) > (m + r(m))^{2k}/2^{2k}$. Now, given x, we non-uniformly form the new input $x1^{r(m)}$ and run $D_{m+r(m)}$ on this input. The resulting circuit decides SAT_m correctly and has size at most $O((m + r(m))^k)$, by the assumption on size of $\{D_n\}$. For large enough m, $O((m + r(m))^k) < (m + r(m))^{2k}/2^{2k}$, which implies that for infinitely many m, SAT_m has circuits of size less than $Ckt(\text{SAT}_m)$ - a contradiction. □

A corollary of Theorem 11 is that if NP doesn't have circuits of size $O(n^k)$, then $\mathsf{NP}/1 \cap \mathsf{SIZE}(\mathsf{poly})$ doesn't have circuits of size $O(n^k)$. This follows since $\mathsf{ONP}/1 \subseteq \mathsf{NP}/1 \cap \mathsf{SIZE}(\mathsf{poly})$. In fact, this kind of collapse result, showing that a fixed polynomial circuit lower bound for a class implies a fixed polynomial circuit lower bound for a language in the class with polynomial-size circuits, holds much more generally, for *any* complexity measure satisfying a certain natural condition. This condition corresponds to "closure under deterministic transductions" as defined by van Melkebeek and Pervyshev [vMP06], but rather than state it formally, we just observe that our proof works for any reasonable complexity class for which we wish to show a circuit lower bound. The proof abstracts out the translation argument in the proof of Theorem 11.

Theorem 12. *Let* C *be a complexity class such as* NP, P^{NP}, MA, BPP *or* PP. *If* C *does not have circuits of size* $O(n^k)$, *then* $\mathsf{C}/1 \cap \mathsf{SIZE}(\mathsf{poly})$ *does not have circuits of size* $O(n^k)$.

Proof. Let $L' \in \mathsf{C}$ be a language such that L' does not have circuits of size $O(n^k)$. We define a padded language L'' such that L'' in $\mathsf{C}/1 \cap \mathsf{SIZE}(\mathsf{poly})$ and L'' does not have circuits of size $O(n^k)$. L'' is defined from L' in exactly the same way as the language L is defined from SAT in the proof of Theorem 11.

$$L'' = \{x1^r | x \in L', r \text{ is a power of } 2, r \geqslant |x|, Ckt(L'_{|x|}) \leqslant (|x| + r)^{2k}\}$$

The proof that $L'' \notin \mathsf{SIZE}(n^k)$ is exactly as in the proof of Theorem 11. For the upper bound, we define a CTIME machine M with one bit of advice accepting L''. Given an input y of length n, M first decomposes y as $x1^r$, where r is a power of 2 and $r \geqslant |x|$, if such a decomposition is possible. If not, M rejects. If such a decomposition exists, it uniquely determines $|x|$ and r. The bit of advice just specifies if $Ckt(L'_{|x|}) \leqslant (|x| + r)^{2k}$. If yes, M simulates the CTIME machine for L'' on x, accepting iff the simulated machine does. If not, M rejects.

If CTIME is able to simulate deterministic time, as is the case for all the complexity classes in the statement of the theorem, then $L'' \in \mathsf{C}/1$, since every stage of the process above, including the simulation of the machine for L', can be implemented in polynomial time. Also, just by using the optimal circuits for L' to decide L'' on appropriately padded inputs, it follows that L'' has polynomial size circuits, in fact circuits of size $O(n^{2k})$. □

Theorem 12 can be stated as an *equivalence* for the polynomial-time versions of syntactic measures.

Corollary 13. *Let* CTIME *be a syntactic measure, and* C *be the polynomial-time version of that measure, such as* $\mathsf{NP}, \mathsf{P}^{\mathsf{NP}}$ *or* PP. C *does not have circuits of size* n^k *iff* $\mathsf{C}/1 \cap \mathsf{SIZE}(\mathsf{poly})$ *does not have circuits of size* n^k.

The forward implication in Corollary 13 follows from Theorem 12 , and the backward implication from Proposition 1.

We next consider the question of what space complexity is required of languages for which we would like to prove circuit lower bounds. The proof of Theorem 8 yields an n^k size lower bound for a language decided by a Σ_2 machine using polynomial time and $n^k\mathsf{polylog}(n)$ space. Can we show such a lower bound for a language with a smaller space requirement? We prove that an n^k size circuit lower bound for a language in the polynomial hierarchy which can be decided in space $n^{k(1-\Omega(1))}$ would imply non-trivial circuit lower bounds for NP, and is therefore likely to be hard.

We require two lemmas. The first one, due to Nepomnjascii [Nep70] and Reischuk [Rei90], shows that a language decidable in the polynomial hierarchy with sublinear space, can be decided much more efficiently with extra alternations.

Lemma 14. *Given constant $\beta > 0$ and integer $a > 0$, there is an integer b such that $\Sigma_a - \mathsf{TISP}(\mathsf{poly}(n), n^{1-\beta}) \subseteq \Sigma_b - \mathsf{TIME}(O(n))$.*

The second lemma is a collapse lemma showing that circuit lower bounds in the linear-time hierarchy imply circuit lower bounds for non-deterministic linear time.

Lemma 15. *If $\Sigma_b - \mathsf{TIME}(O(n)) \not\subseteq \mathsf{SIZE}(n^k)$, then $\mathsf{NTIME}(O(n)) \not\subseteq \mathsf{SIZE}(n^{k^{1/c}})$ for any $c > b$.*

Proof. The proof is by induction. We show that for each $i > 0$, if $\mathsf{NTIME}(O(n)) \subseteq \mathsf{SIZE}(n^l)$, then $\Sigma_i - \mathsf{TIME}(O(n)) \subseteq \mathsf{SIZE}((n\mathsf{polylog}(n))^{l^i})$.

The base case is trivial. For the inductive case, suppose $\Sigma_i - \mathsf{TIME}(O(n)) \subseteq \mathsf{SIZE}((n\mathsf{polylog}(n))^{l^i})$. Let $L \in \Sigma_{i+1} - \mathsf{TIME}(O(n))$. Then there is a relation $R(x, y)$ such that $R \in \Pi_i - \mathsf{TIME}(O(n))$, and $x \in L$ iff $\exists^{O(n)} y R(x, y)$. By assumption, $R \in \mathsf{SIZE}((n\mathsf{polylog}(n))^{l^i})$, where $n = |x|$ (since $|y$ is linear in $|x|$). Thus, $L \in \mathsf{NSIZE}((n\mathsf{polylog}(n))^{l^i})$. Since $\mathsf{NTIME}(O(n)) \subseteq \mathsf{SIZE}(n^l)$, Cook's theorem gives that $\mathsf{NSIZE}(O(n)) \subseteq \mathsf{SIZE}(n^l\mathsf{polylog}(n))$, and by padding, $\mathsf{NSIZE}((n\mathsf{polylog}(n))^{l^i}) \subseteq \mathsf{SIZE}((n\mathsf{polylog}(n))^{l^{i+1}})$. Thus $L \in \mathsf{SIZE}((n\mathsf{polylog}(n))^{l^{i+1}})$, which finishes the induction. □

Theorem 16. *If there is an integer $a > 0$ and a constant $\gamma > 0$ such that $\Sigma_a - \mathsf{TISP}(\mathsf{poly}(n), n^{k-\gamma}) \not\subseteq \mathsf{SIZE}(n^k)$, then $\mathsf{NTIME}(O(n))$ does not have linear-size circuits.*

Proof. Assume $\Sigma_a - \mathsf{TISP}(\mathsf{poly}(n), n^{k-\gamma}) \not\subseteq \mathsf{SIZE}(n^k)$. Then, by translation, there is $\delta > 0$ such that $\Sigma_a - \mathsf{TISP}(\mathsf{poly}(n), n^{1-\beta}) \not\subseteq \mathsf{SIZE}(n^{1+\delta})$, where $\beta = \gamma/k - \delta$. By Lemma 14, this implies there is a constant b such that $\Sigma_b - \mathsf{TIME}(O(n)) \not\subseteq \mathsf{SIZE}(n^{1+\delta})$. Applying Lemma 15 with $k = 1 + \delta$, we get that $\mathsf{NTIME}(O(n)) \not\subseteq \mathsf{SIZE}(n)$. □

Note that superlinear circuit lower bounds are not known for NP, let alone for $\mathsf{NTIME}(O(n))$.

4 Relativized Circuit Upper Bound

Theorem 17. *There exists a an oracle relative to which $\oplus\mathsf{P}$ is contained in $\mathsf{SIZE}(n^k)$ for some constant k.*

Proof. Beigel, Buhrman and Fortnow [BBF98] created an oracle relative to which

$$\mathsf{P} = \oplus\mathsf{P} \text{ and } \mathsf{NP} = \mathsf{EXP}.$$

We will use the same oracle.

By Valiant-Vazirani [VV86], NP is in $\mathsf{BPP}^{\oplus\mathsf{P}}$. Since we have $\oplus\mathsf{P} = \mathsf{P}$ and $\mathsf{BPP} \subseteq \mathsf{SIZE}(\mathsf{poly})$ we have $\mathsf{EXP} = \mathsf{NP} \subseteq \mathsf{SIZE}(\mathsf{poly})$.

Under a standard padding argument $\oplus\mathsf{P} = \mathsf{P}$ implies $\oplus\mathsf{E}$ and so we have

$$\oplus\mathsf{E} \subset \mathsf{EXP} \subset \mathsf{SIZE}(\mathsf{poly}).$$

Let L be a linear-time complete set for $\oplus\mathsf{E}$. There is some k such that L is in $\mathsf{SIZE}(n^k)$ and since L is linear-time complete we have

$$\oplus\mathsf{P} \subset \oplus\mathsf{E} \subset \mathsf{SIZE}(n^k).$$

□

By an analysis of the proof of Beigel, Buhrman and Fortnow [BBF98], we can show that $\oplus\mathsf{P} \subseteq \mathsf{SIZE}(n^4)$ relative to their oracle. With a more careful reworking of their proof we can show $\oplus\mathsf{P} \subseteq \mathsf{SIZE}(n^2)$ for a relativized world (proof omitted).

Vinodchandran [Vin05] shows that $\mathsf{PP} \not\subseteq \mathsf{SIZE}(n^k)$ for any fixed k and his proof does not relativize. The nonrelativizing part of Vinodchandran's proof uses the fact that $\mathsf{P}^{\mathsf{PP}} \subseteq \mathsf{SIZE}(\mathsf{poly})$ implies $\mathsf{P}^{\mathsf{PP}} \subseteq \mathsf{MA}$ (see [BFL91a]). Since $\mathsf{P}^{\oplus\mathsf{P}} \subseteq \mathsf{SIZE}(\mathsf{poly})$ also implies $\mathsf{P}^{\oplus\mathsf{P}} \subseteq \mathsf{MA}$ [FF93] perhaps one can prove a similar circuit lower bound for $\oplus\mathsf{P}$ despite Theorem 17?

However $\mathsf{P}^{\oplus\mathsf{P}} \subseteq \mathsf{MA}$ (and the much stronger $\mathsf{EXP} = \mathsf{BPP}$) relative to the Beigel-Buhrman-Fortnow oracle. We have the following contrasting results.

Corollary 18 (Vinodchandran). *Relative to all oracles, if* $\mathsf{P}^{\mathsf{PP}} \subseteq \mathsf{MA}$ *then* PP *is not contained in* $\mathsf{SIZE}(n^k)$ *for any fixed* k*.*

Corollary 19 (Theorem 17). *Relative to some oracle,* $\mathsf{P}^{\oplus\mathsf{P}} \subseteq \mathsf{MA}$ *but* $\oplus\mathsf{P}$ *is contained in* $\mathsf{SIZE}(n^2)$*.*

Thus to prove $\oplus\mathsf{P} \not\subseteq \mathsf{SIZE}(n^2)$ one would need nonrelativizing techniques beyond those used by Vinodchandran.

5 Circuit Lower Bounds

In this section, we show fixed polynomial circuit lower bounds for circuit models that are more powerful than conventional deterministic circuits. Vinodchandran [Vin05] showed that PP does not have circuits of size n^k for any k. We show a much stronger lower bound for P^{PP}.

Theorem 20. *For any* $k > 0$, $\mathsf{P}^{\mathsf{PP}} \not\subseteq \oplus\mathsf{SIZE}(n^k)$*.*

Proof. Theorem 8 gives that $\Sigma_2 \not\subseteq \mathsf{SIZE}(n^k)$. This theorem relativizes, hence we have that $\Sigma_2^{\oplus\mathsf{P}} \not\subseteq \oplus\mathsf{SIZE}(n^k)$. By Toda's theorem [Tod89], $\Sigma_2^{\oplus\mathsf{P}} \subseteq \mathsf{BPP}^{\oplus\mathsf{P}} \subseteq \mathsf{P}^{\mathsf{PP}}$, which yields our result. □

It would be interesting to strengthen Theorem 20 to derive the same lower bound for PP. Unlike Theorem 20, this cannot be done in a relativizing way, since Aaronson [Aar05] constructed an oracle relative to which PP has linear size circuits.

We can show a stronger version of Theorem 20 using a similar proof:

Theorem 21. *For any* $k > 0$ *and* $a > 0$, P^{PP} *does not have circuits of size* n^k *with* $\Sigma_a^{\oplus\mathsf{P}}$ *oracle gates.*

We can also show a lower bound for PEXP against fixed polynomial size advice by combining a diagonalization technique with closure properties of PP.

Theorem 22. *For any* $k > 0$, $\mathsf{PEXP} \not\subseteq \mathsf{PE}/n^k$*.*

Proof. Let $M_1, M_2, \ldots$ be an enumeration of relativized PE machines. For each M_i pick a sufficiently large input length n. By a standard counting argument there must be a circuit of size n^{k+1} that is not computed by M_i using n^k bits of advice. We can compute the lexicographically least such circuit in $\mathsf{EXP}_{||}^{\mathsf{PEXP}}$ by asking for all advice strings a and inputs x whether $M_i(x)$ accepts using advice a and then searching in EXP for the first circuit that differs. Fortnow and Reingold [FR96] show that $\mathsf{P}_{||}^{\mathsf{PP}} = \mathsf{PP}$ and thus $\mathsf{EXP}_{||}^{\mathsf{PEXP}} = \mathsf{PEXP}$. □

References

[Aar05] Scott Aaronson. Oracles are subtle but not malicious. *Electronic Colloquium on Compuational Complexity*, 12(40), 2005.

[AKSS95] Vikraman Arvind, Johannes Kobler, Uwe Schoning, and Rainer Schuler. If NP has polynomial-size circuits, then MA=AM. *Theoretical Computer Science*, 137(2):279–282, 1995.

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

[BBF98] R. Beigel, H. Buhrman, and L. Fortnow. NP might not be as easy as detecting unique solutions. pages 203–208. ACM, New York, 1998.

[BCG+96] Nader Bshouty, Richard Cleve, Ricard Gavalda, Sampath Kannan, and Christino Tamon. Oracles and queries that are sufficient for exact learning. *Journal of Computer and System Sciences*, 52(2):268–286, 1996.

[BFL91a] L. Babai, L. Fortnow, and C. Lund. Non-deterministic exponential time has two-prover interactive protocols. 1(1):3–40, 1991.

[BFL91b] László Babai, Lance Fortnow, and Carsten Lund. Non-deterministic exponential time has two-prover interactive protocols. *Computational Complexity*, 1:3–40, 1991.

[BFNW93] László Babai, Lance Fortnow, Noam Nisan, and Avi Wigderson. BPP has subexponential time simulations unless EXPTIME has publishable proofs. *Computational Complexity*, 3(4):307–318, 1993.

[BFT98] Harry Buhrman, Lance Fortnow, and Thomas Thierauf. Nonrelativizing separations. In *Proceedings of 13th Annual IEEE Conference on Computational Complexity*, pages 8–12, 1998.

[Cai01] Jin-Yi Cai. $s_2^p \subseteq zpp^{NP}$. In *Proceedings of the 42nd Annual Symposium on Foundations of Computer Science*, pages 620–629, 2001.

[CR06] Venkat Chakaravarthy and Sambuddha Roy. Oblivious symmetric alternation. In *Proceedings of Symposium on Theoretical Aspects of Computer Science*, pages 230–241, 2006.

[FF93] J. Feigenbaum and L. Fortnow. On the random-self-reducibility of complete sets. *SIAM Journal on Computing*, 22:994–1005, 1993.

[FR96] L. Fortnow and N. Reingold. PP is closed under truth-table reductions. *Information and Computation*, 124(1):1–6, 1996.

[IKW02] Russell Impagliazzo, Valentine Kabanets, and Avi Wigderson. In search of an easy witness: Exponential time vs. probabilistic polynomial time. *Journal of Computer and System Sciences*, 65(4):672–694, 2002.

[Kan82] Ravi Kannan. Circuit-size lower bounds and non-reducibility to sparse sets. *Information and Control*, 55(1):40–56, 1982.

[KW98] Johannes Kobler and Osamu Watanabe. New collapse consequences of np having small circuits. *SIAM Journal on Computing*, 28(1):311–324, 1998.

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

[Nep70] V. Nepomnjascii. Rudimentary predicates and turing calculations. *Soviet Mathematics - Doklady*, 11(6):1462–1465, 1970.

[Rei90] Rudiger Reischuk. *Einfuhrung in die Komplexitatstheorie*. Teubner, 1990.

[San06] R. Santhanam. Circuit lower bounds for Merlin-Arthur classes, 2006. Manuscript.

[Sha92] Adi Shamir. IP = PSPACE. *Journal of the Association for Computing Machinery*, 39(4):869–877, 1992.

[Tod89] Seinosuke Toda. On the computational power of PP and $\oplus P$. In *30th Annual IEEE Symposium on Foundations of Computer Science*, pages 514–519, 1989.

[Vin05] Variyam Vinodchandran. A note on the circuit complexity of pp. *Theoretical Computer Science*, 347(1-2):415–418, 2005.

[vMP06] Dieter van Melkebeek and Konstantin Pervyshev. A generic time hierarchy for semantic models with one bit of advice. In *Proceedings of 21st Annual IEEE Conference on Computational Complexity*, pages 129–144, 2006.

[VV86] L. Valiant and V. Vazirani. NP is as easy as detecting unique solutions. *Theoretical Computer Science*, 47:85–93, 1986.

ECCC ISSN 1433-8092

http://eccc.hpi-web.de/