

# Better Time-Space Lower Bounds for SAT and Related Problems

Ryan Williams\*  
Computer Science Department  
Carnegie Mellon University  
Pittsburgh, PA 15213  
ryanw@cs.cmu.edu

## Abstract

*We make several improvements on time lower bounds for concrete problems in NP and PH.*

- 1. We present an elementary technique based on “indirect diagonalization” that uniformly improves upon the known nonlinear time lower bounds for nondeterminism and alternating computation, on both sublinear ( $n^{o(1)}$ ) space RAMs and sequential worktape machines with random access to the input. We obtain better lower bounds for SAT, as well as all NP-complete problems that have efficient reductions from SAT, and  $\Sigma_k$ -SAT, for constant  $k \geq 2$ . For example, SAT cannot be solved by random access machines using  $n^{\sqrt{3}}$  time and  $n^{o(1)}$  space. The technique is a natural inductive approach, for which previous work is essentially its base case.*
- 2. We show how indirect diagonalization can also yield time-space lower bounds for computation with bounded nondeterminism. One corollary is that for all  $k$ , there exists a constant  $c_k > 1$  such that satisfiability of Boolean circuits with  $n$  inputs and  $n^k$  gates cannot be solved in deterministic time  $n^{k \cdot c_k}$  and  $n^{o(1)}$  space.*

## 1 Introduction

In this paper, we study the power of indirect diagonalization to give class separations and lower bounds on explicit problems in NP and the polynomial hierarchy, deriving several improvements on existing lower bounds and some brand new bounds as well.

---

\*Supported in part by the NSF ALADDIN Center (NSF Grant No. CCR-0122581).

Put bluntly, a separation by “indirect diagonalization” is a proof-by-contradiction simulation. Suppose we wish to show  $\mathcal{C} \not\subseteq \mathcal{D}$ , for classes  $\mathcal{C}$  and  $\mathcal{D}$ . An indirect diagonalization argument begins by assuming  $\mathcal{C} \subseteq \mathcal{D}$ . If sufficiently strong, this assumption allows us to derive a sequence of new complexity class inclusions. After some iterations, the new inclusions become so wonderful that they are not only unlikely but are also provably impossible, contradicting a known separation, *e.g.*, a time hierarchy theorem.

One limitation to this sort of attack is that, at present, there are not too many known class separations to begin with, so a number of strong assumptions are sometimes necessary to reach a contradiction. The major goal of our work is to investigate how one might circumvent this difficulty, by employing a host of known class separations in some sophisticated way. Loosely speaking, one family of prior lower bound approaches for nondeterministic time (in general, alternating time) is of the following kind:

1. Assume (for contradiction) that for some  $k \geq 1$ ,  $\Sigma_k \text{TIME}[n]$  is in class  $\mathcal{D}$  whose languages are characterized by some deterministic machine model running in time  $n^c$ .
2. Prove that  $\mathcal{D}$ -machines in time  $t$  can be “sped up” to lie in  $\Sigma_\ell \text{TIME}[t^{1-\varepsilon}]$  for some  $\varepsilon > 0$  and some  $\ell \geq k$ .
3. Prove using (1) that a sufficiently large number of the  $\ell$  alternations in the sped-up computation of (2) can be “removed”, at the cost of a small “slowdown” in time (the slowdown will be exponential in  $c$ ).
4. If the amount of speedup in (2) sufficiently exceeds the amount of slowdown in (3), conclude a contradiction to some known time hierarchy theorem.

Introduced in a paper by Kannan [10], this proof strategy has been quite successful, leading to a number of lower bounds on nondeterminism and alternation in several machine models [12, 6, 9, 7, 18]. Here, we propose a strategy that, given an assumption like (1), inductively derives

a countably infinite number of inclusions to obtain a contradiction, where each successive inclusion uses all of the (finitely many) previous ones.

To illustrate, in the RAM lower bounds we suppose that  $\text{NTIME}[n] \subseteq \text{DTISP}[n^c, n^{o(1)}]$  for some  $c \in (1, 2)$  and in turn we derive an inclusion of the form

$$\Sigma_2 \text{TIME}[n] \subseteq \Pi_2 \text{TIME}[n^{f_2(c)}],$$

where  $f_2$  is a function of  $c$ . If  $f_2(c) < 1$ , then the inclusion contradicts a known separation (cf. Theorem 2 in the following section). Otherwise, we may assume this inclusion as well.<sup>1</sup> We use this inclusion to derive  $\Sigma_3 \text{TIME}[n] \subseteq \Pi_3 \text{TIME}[n^{f_3(c)}]$ , where  $f_3(c) < f_2(c)$  for appropriate  $c$ . Again if  $f_3(c) < 1$  we would be done, otherwise we proceed to get an inclusion for  $\Sigma_4 \text{TIME}[n]$ ,  $\Sigma_5 \text{TIME}[n]$ , etc. Our construction and choice of  $c$  will ensure that the sequence  $f_2(c), f_3(c), f_4(c), \dots$  is monotonically decreasing, and eventually drops below 1. Moreover, the  $c$  such that the sequence drops below 1 will be larger than the lower bound exponents previously obtained, cf. Table 1.

The most dramatic improvements occur with higher levels of the polynomial hierarchy: for example, with  $\Sigma_2$ -SAT the lower bound goes from  $n^2$  time to  $n^{2.761}$ . Note  $1.6616^2 \approx 2.761$ ; in fact, the exponent in the  $\Sigma_2$ -SAT lower bound for deterministic RAMs is exactly the square root of the exponent in the bound for co-nondeterministic RAMs. Similarly, the exponent in the lower bound for  $\Sigma_2$ -SAT on deterministic off-line machines is the square of the exponent for SAT.

## 2 Preliminaries

### 2.1 Notation and the model

We will often use  $\varepsilon$  here and there to denote (as is standard) a non-zero quantity that is sufficiently small for the current context. As is typical with most such works, we implicitly assume floors and ceilings are applied to fractions wherever appropriate.

We assume familiarity with basic notions such as alternation [2], and classes specifying resource bounds such as  $\text{DTIME}[t]$ ,  $\text{NTIME}[t]$ ,  $\text{SPACE}[s]$ ,  $\text{DTISP}[t]$  (simultaneous deterministic time and space), and  $\Sigma_k \text{TIME}[t]$  and  $\Pi_k \text{TIME}[t]$  (time with  $k$  alternations;  $\Sigma$  denotes starting in an existential state, whereas  $\Pi$ -machines start in a universal state). Our default machine model will be random access Turing machines. When we specify a class without further qualification, we will be referring to classes defined with respect to this model. By “random access Turing machine”,

<sup>1</sup>In fact, observe  $e_2 = 1$  would imply  $\Sigma_k \text{TIME}[n] = \Sigma_2 \text{TIME}[n]$  for all  $k \geq 2$ ; it turns out this would be also sufficient for a contradiction.

we mean Turing machines with a read-only input tape, a full-access work tape, and two write-only index tapes (one for input, one for work). To access the  $i$ th cell of the input or work tape, one writes  $i$  to the respective index tape; hence an arbitrary access of a tape with  $t$  cells takes  $O(\log t)$  time. After the  $i$ th cell is accessed, the respective index tape is reset to blanks.

A word of apology to the precise reader: we will say “ $k$  alternations” to refer to the  $k$  quantifier blocks of an alternating machine in  $\Sigma_k$  or  $\Pi_k$ , which really only alternate  $k - 1$  times from one quantifier mode to another.

### 2.2 Existing tools

We will use the fact that satisfiability of Boolean formulas (specified in conjunctive normal form) is complete under very tight reductions for a small complexity class. Define  $\text{NQL} := \bigcup_{c>0} \text{NTIME}[n(\log n)^c] = \text{NTIME}[n \cdot \text{poly}(\log n)]$ . Hence  $\text{NQL}$  means “nondeterministic quasi-linear time”.

**Theorem 1** (FOLLOWS FROM SCHNORR [17], COOK [5], GUREVICH AND SHELAH [8], TOURLAKIS [18])

*Assuming random access to the input, SAT is NQL-complete, under reductions in quasi-linear time and  $O(\log n)$  space simultaneously, for both multi-tape and random access machine models. Moreover, each bit of the reduction can be computed in  $O((\log n)^{O(1)})$  time.*

This theorem has a corollary significant for our purposes. Let  $\text{DTIME}[t]$  in the following be characterized by some deterministic machine model running in time  $O(t)$ , for which the machines have random access to the input.

**Corollary 1** *If  $\text{NTIME}[n] \not\subseteq \text{DTIME}[t]$ , then there is a  $c > 0$  such that  $\text{SAT} \notin \text{DTIME}[t(\log n)^c]$ .*

That is, if one can show  $\text{NTIME}[n]$  is not doable in  $t$  time, then one can name an *explicit problem* (SAT) not in  $t$  time, modulo polylogarithmic factors. Thus our proofs will attempt to establish that  $\text{NTIME}[n]$  is not in some deterministic class with time bound  $n^{1+\varepsilon}$ , implying that SAT is not solvable in  $n^{1+\varepsilon-o(1)}$  time with respect to that class.

Furthermore, as observed by Van Melkebeek and Raz, the results of this work apply to any problem  $\Pi$  such that SAT reduces to  $\Pi$  under highly efficient reductions. Examples of such problems include Vertex Cover, Independent Set, Travelling Salesman, 3-SAT, and MAX-2-SAT. (The typical reductions from SAT to these problems use gadgets, where there is an explicit and simple correspondence between each clause of the original formula and each gadget of the reduced instance.)

Problem	Model of computation		
	det. RAM	co-nondet. RAM	det. off-line TM (one worktape)
SAT	$n^{1.7327}$ ( $n^{1.618}$ )	$n^{1.337}$ ( $n^{1.324}$ )	$n^{1.268}$ ( $n^{1.224}$ )
$\Sigma_2$ -SAT	$n^{2.761}$ ( $n^2$ )	$n^{1.6616}$	$n^{1.609}$ ( $n^{1.5}$ )
$\Sigma_3$ -SAT	$n^{3.812}$ ( $n^3$ )	$n^{2.761}$	$n^{1.726}$
$\Sigma_{100}$ -SAT	$n^{100.99}$ ( $n^{100}$ )	$n^{99.98}$	$n^{1.99}$

**Table 1. Time lower bounds of this paper.** Of course, for RAMs, the given bounds hold assuming at most  $n^{o(1)}$  worktape is used. The previous bounds are in parentheses. Each of them either appear in [7] (the RAM bounds), [13] (the TM bounds), or can be derived from that work.

**Corollary 2** (CF. VAN MELKEBEEK AND RAZ [13]) *Our lower bounds apply to any problem  $\Pi$  such that SAT reduces to  $\Pi$  under reductions computable in  $n^{1+o(1)}$  time, where any particular bit of the reduction is computable in  $n^{o(1)}$  time.*

We will also use some well-known separation results. The following uses the fact that a random-access machine using  $k$  alternations in time  $t$  can be simulated by a two-tape machine using  $k$  alternations in time  $O(t)$ , found in Chandra and Stockmeyer’s original conference paper on alternation [3].

**Theorem 2** (“NO COMPLEMENTARY SPEEDUP”) *For all  $k \geq 1$  and time constructible  $t(n) \geq n$ ,*  
 $\Sigma_k \text{TIME}[t] \not\subseteq \Pi_k \text{TIME}[o(t)].$

We call it the “No Complementary Speedup” Theorem, as it intuitively says that a bounded-alternation machine cannot be sped-up by a “complementary” machine with the same number of alternations. Another useful proposition says we can reduce alternations in a computation while slightly increasing the time, provided a close time relationship exists between smaller alternation classes.

**Lemma 1** (ALTERNATIONS FOR TIME LEMMA) *Let  $d > 1$  be rational,  $k$  and  $\ell$  be non-negative integers with  $k > \ell$ , and let  $t(n) \geq n$  be time constructible.*

*If  $\Sigma_\ell \text{TIME}[n] \subseteq \Pi_\ell \text{TIME}[n^d]$ , then*

- $\Sigma_k \text{TIME}[t] \subseteq \Sigma_{k-1} \text{TIME}[t^d]$ , and
- $\Pi_k \text{TIME}[t] \subseteq \Pi_{k-1} \text{TIME}[t^d]$ .

**Proof.** Let  $M$  be a  $\Sigma_k$  machine with  $O(t)$  runtime. W.l.o.g., for each  $i \in [k]$  we may assume that the state of  $M$  immediately prior to the  $i$ th alternation (switch between modes) is a special state  $q_i$ . On an input  $x$ , consider the set  $S$  of nodes in  $M(x)$ ’s configuration tree just before the  $k - \ell^{\text{th}}$  alternation occurs. (Note we can easily recognize membership in  $S$  at runtime, by the assumption.) By the hypothesis of the theorem, we can replace each  $\ell$ -alternation computation subtree

starting at a node of  $S$  with an  $\ell$ -alternation computation tree that begins with the opposite quantifier. The input to the nodes of  $S$  (the current configuration) is of size  $O(t)$ , and the original runtime of a computation subtree rooted at such a node was  $O(t)$ . Hence by the hypothesis, each new computation subtree takes  $O(t^d)$  time, and our replacement reduces the number of alternations overall by 1.  $\square$

The following simulation lemma will also be used, the proof of which we defer to the appendix.

**Lemma 2** (FORTNOW AND VAN MELKEBEEK [7], THEOREM 5.1) *For every natural number  $k \geq 2$ , and time constructible  $t$  and space constructible  $s$ ,*

$$\text{DTISP}[t, s] \subseteq \Sigma_k \text{TIME}[(ts^{k-1})^{1/k}].$$

**Remark 1** *Note that  $\text{DTISP}[t, s] \subseteq \Pi_k \text{TIME}[(ts^{k-1})^{1/k}]$  follows immediately from the proof.*

### 3 Intuition

It should be helpful for us to first make a few high-level remarks on how our method works. If nondeterministic time  $t$  can be simulated in deterministic time  $t^c$ , then it is straightforward that  $\Sigma_k$  time  $t$  is simulated in  $\Sigma_{k-1}$  time  $t^c$ . The fundamental observation behind our results is that, if we further assume nondeterministic time  $t$  can be efficiently simulated by deterministic time  $t^c$  in space  $t^{o(1)}$ , this not only implies that  $\Sigma_k$  time  $t$  can be efficiently simulated in  $\Sigma_{k-1}$  but also that the simulation time is *faster* than  $t^c$ , if  $c$  is sufficiently small. Moreover, as  $k$  increases, the implied simulation gets faster. For a concrete example of this, Lip-ton and Viglas showed  $\text{NTIME}[n] \not\subseteq \text{DTISP}[n^{\sqrt{2}-\varepsilon}, n^{o(1)}]$ , by employing a chain of reasoning akin to the following: assume not, then by the machinery presented in the previous section,

$$\begin{aligned} \Sigma_2 \text{TIME}[n] &\subseteq \text{NTIME}[n^{\sqrt{2}-\varepsilon}] \\ &\subseteq \text{DTISP}[n^{(\sqrt{2}-\varepsilon)^2}, n^{o(1)}] \subseteq \Pi_2 \text{TIME}[n^{\frac{(\sqrt{2}-\varepsilon)^2 + o(1)}{2}}] \\ &\subseteq \Pi_2 \text{TIME}[o(n)], \end{aligned}$$

a contradiction.<sup>2</sup> (Note that our development in later sections will simply drop the  $o(1)$  terms from derivations to keep things clean.) What if  $c \geq \sqrt{2}$ ? Then the resulting derivation

$$\Sigma_2 \text{TIME}[n] \subseteq \Pi_2 \text{TIME}[n^{\frac{c^2}{2}+o(1)}]$$

is not yet a contradiction, but it is at the very least a lemma that  $\Sigma_k \text{TIME}[n] \subseteq \Sigma_{k-1} \text{TIME}[n^{\frac{c^2}{2}+o(1)}]$  for all  $k \geq 3$ . Provided that  $c < 2$ , this is indeed stronger than  $\Sigma_k \text{TIME}[n] \subseteq \Sigma_{k-1} \text{TIME}[n^c]$ . This lemma can then be used to get an even tighter inclusion for  $\Sigma_3$  in  $\Pi_3$ , in particular

$$\begin{aligned} \Sigma_3 \text{TIME}[n] &\subseteq \Sigma_2 \text{TIME}[n^{\frac{c^2}{2}+o(1)}] \\ &\subseteq \text{DTISP}[n^{\frac{c^2}{2} \cdot c^2 + o(1)}, n^{o(1)}] \subseteq \Pi_3 \text{TIME}[n^{\frac{c^4}{8} + o(1)}]. \end{aligned}$$

An inductive approach naturally arises: derive increasingly better  $\Pi_k$  simulations of  $\Sigma_k$  using the previous simulations obtained, and take  $c$  to be the largest constant that still implies  $\Pi_k \text{TIME}[n] \subseteq \Sigma_k \text{TIME}[o(n)]$  for some  $k$ . This particular attack turns out to yield better lower bounds than previous approaches, as we will see throughout the paper.

## 4 New lower bound for SAT and related NP-complete sets

Initiated in work of Fortnow [6] in 1997, an intriguing thread of lower bound research has opened up, with roots going back to Kannan [11], which essentially seeks to prove a “poor man’s” version of  $L \neq \text{NP}$  (or even,  $\text{SC} \neq \text{NP}$ , recall  $\text{SC} = \text{DTISP}[n^{O(1)}, (\log n)^{O(1)}]$ ). More precisely, while it is well-known that  $L \neq \text{NP}$  is equivalent to  $\text{NTIME}[n] \not\subseteq \text{DTISP}[n^k, \log n]$  for all  $k \geq 1$ , proving such a large lower bound on nondeterministic linear time (and therefore, SAT) appears quite difficult and still out of reach.

Nevertheless, it is still interesting to ask what we *can* prove about the matter—to find the largest  $k$  for which the separation provably holds. Of course, when one starts to consider fixed time bounds, the model of computation becomes a possible issue. We use the robust random access Turing machine model discussed in Section 2.1, which is time-equivalent to other random access models within polylogarithmic factors.

Lipton and Viglas [9] found a method employing alternation and a variation on the technique in Savitch’s theorem to obtain the lower bound

$$\text{NTIME}[n] \not\subseteq \text{DTISP}[n^{\sqrt{2}-\varepsilon}, n^{o(1)}].$$

<sup>2</sup>Actually, Lipton and Viglas’ original argument worked by way of a different separation, namely  $\text{NTIME}[t] \not\subseteq \text{NTIME}[t^{1-\varepsilon}]$ . This is a major reason why our method was not discovered before.

By Corollary 1, this implies Satisfiability cannot be solved in  $n^{\sqrt{2}-\varepsilon}$  time and sublinear (*i.e.*  $n^{o(1)}$ ) space, for all  $\varepsilon > 0$ . The best lower bound known prior to our work was  $n^{\phi-\varepsilon}$  time and  $n^{o(1)}$  space by Fortnow and Van Melkebeek [7] in 2000, where  $\phi \approx 1.618$  is the golden ratio. We shall employ tools developed in that and previous work with an inductive argument to improve the lower bound to  $n^{1.6616\dots}$ . This inductive strategy is strictly different from (in a sense, *orthogonal to*) one used by Fortnow and Van Melkebeek. We then use a second inductive argument similar to Fortnow and Van Melkebeek’s to boost this bound to  $n^{1.7327}$ , which is slightly larger than  $n^{\sqrt{3}}$ .

Note these exponents were derived using numerical computation. We do not yet know closed-form expressions for them, and our experience of trying to obtain one suggests that an explicit formula for these constants are beyond our current analytical understanding. Namely, it appears that any attempt to characterize the constants in an interesting way requires the use of polylogarithm functions (those of the form  $f(x, k) = \sum_{i=1}^{\infty} \frac{x^i}{i^k}$ ), which are infamously difficult to compute exactly.

### 4.1 First SAT Lower Bound

We begin by showing the  $n^{1.6616}$  lower bound. To ensure that our method is fully elucidated to the reader, our application of it will be more detailed here than in subsequent sections.

Define  $f : \mathbb{N} \rightarrow \mathbb{N}$  as  $f(k) := \prod_{j=1}^{k-1} (1 + 1/j)^{1/2^j}$ .

**Theorem 3** *For every  $k \geq 2$  and rational  $c$  such that  $c < f(k)$ ,  $\text{NTIME}[n] \not\subseteq \text{DTISP}[n^c, n^{o(1)}]$ .*

**Corollary 3** *Satisfiability is not solvable by deterministic random access machines using  $n^{f(k)-\varepsilon}$  time and  $n^{o(1)}$  space, for all  $k \geq 2$  and  $\varepsilon > 0$ . The lower bound holds for any NP-complete problem that is (simultaneously) quasi-linear time and polylogspace reducible from satisfiability, where each bit in the reduction is computable in  $n^{o(1)}$  time.*

Before we begin the proof of the theorem, let us first observe some properties of the  $f$  function.

**Lemma 3**  *$f(k)$  is monotone increasing and converges to a value greater than 1.6616.*

**Proof of Lemma 3.** As  $(1 + 1/j)^{1/2^j} > 1$  for all  $j$ , it is evident that  $f(k)$  is monotone increasing. Observe that  $(1 + 1/j)^{1/2^j} \leq \exp(\frac{1}{j \cdot 2^j})$ , so  $f(k) \leq \exp(\sum_{j=1}^{k-1} \frac{1}{j \cdot 2^j})$ . As this sum converges, thus  $f(k)$  converges. Computation of  $f(12)$  suffices to show  $f(k) > 1.6616$ .  $\square$

**Proof of Theorem 3.** By induction. Let  $k'$  be the smallest integer such that  $c < f(k')$ . We first prove the theorem when  $k' = 2$ . Assume for contradiction that

$$\text{NTIME}[n] \subseteq \text{DTISP}[n^c, n^{o(1)}], \quad (*)$$

for rational  $c > 1$ . Define an expression  $e(k)$  in terms of  $c$  inductively:

$$e(1) := 1, \quad e(k) := \frac{c^2}{k} \left( \prod_{i=1}^{k-1} e(i) \right).$$

Inclusion  $(*)$  implies

$$\Sigma_2 \text{TIME}[n] \subseteq \text{DTISP}[n^{c^2}, n^{o(1)}] \subseteq \Pi_2 \text{TIME}[n^{e(2)}], \quad (**)$$

where the last inclusion follows from Lemma 2.

If  $c < 2^{1/2} = (1 + 1/1)^{1/2} = f(2)$ , then  $e(2) < 1$ , inclusion  $(**)$  contradicts the ‘‘No Complementary Speedup’’ Theorem, and this would conclude the base case. Otherwise, observe that  $c \geq 2^{1/2}$  implies  $e(2) \geq 1$ . In fact,

**Claim 1** For all  $i$ ,  $e(i) \geq 1 \iff c \geq f(i)$ .

**Proof of Claim 1.** First, we claim  $e(i) = \frac{c^{2^{i-1}}}{i! \prod_{j=2}^{i-2} (j!)^{2^{i-j-2}}}$  follows from a proof by induction. The denominator can be simplified further to get  $e(i) = \frac{c^{2^{i-1}}}{i \cdot \prod_{j=2}^{i-1} j^{2^{i-j-1}}}$ .

For all  $i$ , let  $c_i$  be the unique number in  $(1, 2)$  such that  $e(i) = 1$  when  $c = c_i$ , i.e.  $(c_i)^{2^{i-1}} = i \cdot \prod_{j=2}^{i-1} j^{2^{i-j-1}}$ . It suffices for us to show that  $c_i = f(i)$ . Observe

$$(c_{i-1})^{2^{i-1}} = ((c_{i-1})^{2^{i-2}})^2 = (i-1)^2 \cdot \prod_{j=2}^{i-2} j^{2^{i-j-1}}$$

by definition of  $c_{i-1}$ . Hence

$$\left( \frac{c_i}{c_{i-1}} \right)^{2^{i-1}} = \frac{i \cdot \prod_{j=2}^{i-1} j^{2^{i-j-1}}}{(i-1)^2 \prod_{j=2}^{i-2} j^{2^{i-j-1}}} = i/(i-1),$$

so  $\frac{c_i}{c_{i-1}} = \left( \frac{i}{i-1} \right)^{1/2^{i-1}}$ . Therefore,

$$\begin{aligned} c_i &= (c_i/c_{i-1})(c_{i-1}/c_{i-2}) \cdots (c_3/c_2)c_2 \\ &= \prod_{j=2}^i \left( 1 + \frac{1}{j-1} \right)^{\frac{1}{2^{j-1}}} = \prod_{j=1}^{i-1} \left( 1 + \frac{1}{j} \right)^{\frac{1}{2^j}} = f(i). \square \end{aligned}$$

From the proof of the above claim it follows that, for a fixed  $c$ ,  $e(i)$  strictly decreases as  $i$  increases. Given Claim 1 and our choice of  $k'$ , we have that  $k'$  is the smallest integer such that  $e(k') < 1$ . Therefore,  $e(i) \geq 1$  for all  $i \leq k' - 1$ , and we may assume the following induction hypothesis.

*Induction Hypothesis:* Assume for all  $i \leq k' - 1$  that  $e(i) \geq 1$  and  $\Sigma_i \text{TIME}[n] \subseteq \Pi_i \text{TIME}[n^{e(i)}]$ .

We now prove the theorem for  $k'$ . The alternations-for-time lemma (Lemma 1) and induction hypothesis imply that  $\Sigma_\ell \text{TIME}[n] \subseteq \Sigma_{\ell-1} \text{TIME}[n^{e(\ell-1)}]$  for  $\ell \in \{2, \dots, k' - 1\}$ . We therefore have by padding (which is possible since each  $e(i) \geq 1$ )

$$\begin{aligned} \Sigma_{k'} \text{TIME}[n] &\subseteq \Sigma_{k'-1} \text{TIME}[n^{e(k'-1)}] \\ &\subseteq \Sigma_{k'-2} \text{TIME}[n^{e(k'-1)e(k'-2)}] \\ &\subseteq \cdots \subseteq \Sigma_2 \text{TIME}[n^{\prod_{i=2}^{k'-1} e(i)}]. \end{aligned}$$

But we also have

$$\begin{aligned} \Sigma_2 \text{TIME}[n^{\prod_{i=2}^{k'-1} e(i)}] &\subseteq \text{NTIME}[n^{c \prod_{i=2}^{k'-1} e(i)}] \\ &\subseteq \text{DTISP}[n^{c^2 \prod_{i=2}^{k'-1} e(i)}, n^{o(1)}] \\ &\subseteq \Pi_{k'} \text{TIME}[n^{\frac{c^2}{k'} \prod_{i=2}^{k'-1} e(i)}] \\ &\subseteq \Pi_{k'} \text{TIME}[n^{e(k')}], \end{aligned}$$

where the penultimate inclusion follows from Lemma 2, and the last inclusion follows by definition of  $e(k')$ . As  $c < f(k')$ , Claim 1 implies  $e(k') < 1$ , therefore the above contradicts the ‘‘No Complementary Speedup’’ Theorem. This finishes the proof of Theorem 3.  $\square$

The mechanics of the above proof also demonstrate a new time-space tradeoff for SAT.

**Corollary 4** If SAT is in  $\text{DTISP}[t, s]$  then  $ts^{k-1} \in \Omega(n^{f(k)})$  for all  $k \geq 2$ . For example, SAT is not in  $n^{1.66}$  time and  $n^{10^{-5}}$  space.

## 4.2 From $n^{1.661}$ to $n^{1.732}$

In the above, the  $\text{DTISP}[t, t^{o(1)}] \subseteq \Pi_k \text{TISP}[t^{1/k+o(1)}]$  inclusion is *unconditional*, whereas all other derived inclusions actually depend on the assumption  $\text{NTIME}[n] \subseteq \text{DTISP}[n^c, n^{o(1)}]$ . Here we show how to use this assumption to get  $\text{DTISP}[n^c, n^{o(1)}] \subseteq \Pi_k \text{TISP}[n^{c/(k+\varepsilon)+o(1)}]$  for some  $\varepsilon > 0$ . This allows us to push the SAT lower bound above  $n^{\sqrt{3}}$ .

**Lemma 4** Let  $c < 2$ . Define  $d_1 := 2$ ,  $d_k := 1 + \frac{d_{k-1}}{c}$ .

If  $\text{NTIME}[n^{2/c}] \subseteq \text{DTISP}[n^2, n^{o(1)}]$ , then

for all  $k \in \mathbb{N}$ ,  $\text{DTISP}[n^{d_k}, n^{o(1)}] \subseteq \Pi_2 \text{TIME}[n^{1+o(1)}]$ .

**Proof.** By induction on  $k$ . The  $k = 1$  case is trivial since  $\text{DTISP}[n^2, n^{o(1)}] \subseteq \Pi_2 \text{TIME}[n^{1+o(1)}]$  follows unconditionally.

Suppose  $\text{NTIME}[n^{2/c}] \subseteq \text{DTISP}[n^2, n^{o(1)}]$  and  $\text{DTISP}[n^{d_k}, n^{o(1)}] \subseteq \Pi_2 \text{TIME}[n^{1+o(1)}]$ . By padding and the inductive hypothesis (note  $d_k \geq 2$  for all  $k$ ) we have

$$\text{NTIME}[n^{d_k/c}] \subseteq \text{DTISP}[n^{d_k}, n^{o(1)}] \subseteq \Pi_2 \text{TIME}[n^{1+o(1)}].$$

Consider a  $\Pi_2$  simulation of  $\text{DTISP}[n^{1+d_k/c}, n^{o(1)}]$  where we only guess  $O(n)$  bits (i.e.  $n^{1-o(1)}$  configurations) in the universal quantifier:

$$\left( \begin{array}{l} \forall \text{ configurations } C_1, \dots, C_{n^{1-o(1)}} \text{ of } M \text{ on } x \\ \text{s.t. } C_{n^{1-o(1)}} \text{ is rejecting} \\ (\exists i \in \{1, \dots, n^{1-o(1)} - 1\}) \\ [C_i \text{ does not lead to } C_{i+1} \text{ in } n^{d_k/c+o(1)} \text{ time}] \end{array} \right)$$

The second and third lines of the above corresponds to an NTIME computation that takes an input of length  $O(n)$  (the input  $x$  plus the list of configurations) and runs in  $n^{d_k/c+o(1)}$  time. Hence it can be replaced with a  $\Pi_2$   $n^{1+o(1)}$ -time computation, i.e. it is equivalent to

$$\left( \begin{array}{l} \forall \text{ configurations } C_1, \dots, C_{n^{1-o(1)}} \text{ of } M \text{ on } x \\ \text{s.t. } C_{n^{1-o(1)}} \text{ is rejecting} \\ (\forall y, |y| = c|x|^{1+o(1)}) (\exists z, |z| = c|z|^{1+o(1)}) \\ [R(C_1, \dots, C_{n^{1-o(1)}}, x, y, z)], \end{array} \right)$$

for some deterministic linear time relation  $R$  and constant  $c > 0$ .

Thus  $\text{DTISP}[n^{d_{k+1}}, n^{o(1)}]$  is in  $\Pi_2 \text{TIME}[n^{1+o(1)}]$ .  $\square$

### Corollary 5

Let  $c \in (1, 2)$ . If  $\text{NTIME}[n^{2/c}] \subseteq \text{DTISP}[n^2, n^{o(1)}]$  then for all  $\varepsilon > 0$ ,  $\text{DTISP}[n^{\frac{c}{c-1}-\varepsilon}, n^{o(1)}] \subseteq \Pi_2 \text{TIME}[n^{1+o(1)}]$ .

(Note  $\varepsilon$  must satisfy  $\frac{c}{c-1} - \varepsilon \geq 1$ ; that is,  $\varepsilon \leq \frac{1}{c-1}$ .)

**Proof.** For any  $c < 2$ , the sequence  $d_k$  is monotone non-decreasing, and converges to  $d_\infty = 1 + \frac{d_\infty}{c}$ , i.e.  $d_\infty = c/(c-1)$ . Hence for all  $\varepsilon > 0$ , there is a finite  $K$  such that  $d_K \geq \frac{c}{c-1} - \varepsilon$ .  $\square$

If we apply the above to our inductive argument from before, we see an interesting result: instead of having Lipton-Viglas'  $n^{\sqrt{2}}$  lower bound as a base case, we now have something resembling Fortnow-Van Melkebeek's  $n^\phi$  lower bound as a base case, with  $\phi$  being the golden ratio. In particular, we know that if  $\text{NTIME}[n] \subseteq \text{DTISP}[n^c, n^{o(1)}]$  then  $c \geq \phi$ , hence  $c^2 \geq c/(c-1)$  and Corollary 5 implies

$$\begin{aligned} \Sigma_2 \text{TIME}[n] &\subseteq \text{DTISP}[n^{c^2}, n^{o(1)}] \\ &\subseteq \text{DTISP}\left[\left(n^{c^2 \cdot \frac{c-1}{c}}\right)^{c/(c-1)+o(1)}, n^{o(1)}\right] \\ &\subseteq \Pi_2 \text{TIME}[n^{c \cdot (c-1)+o(1)}]. \end{aligned}$$

Employing an analogous argument as before and omitting  $o(1)$  factors, we have

$$\begin{aligned} \Sigma_3 \text{TIME}[n] &\subseteq \Sigma_2 \text{TIME}[n^{c \cdot (c-1)}] \\ &\subseteq \text{DTISP}[n^{c^3 \cdot (c-1)}, n^{o(1)}] \subseteq \Pi_3 \text{TIME}[n^{\frac{c^3 \cdot (c-1)}{3}}], \end{aligned}$$

$$\begin{aligned} \Sigma_4 \text{TIME}[n] &\subseteq \Sigma_3 \text{TIME}[n^{\frac{c^3 \cdot (c-1)}{3}}] \\ &\subseteq \Sigma_2 \text{TIME}[n^{\frac{c^4 \cdot (c-1)^2}{3}}] \subseteq \text{DTISP}[n^{\frac{c^6 \cdot (c-1)^2}{3}}, n^{o(1)}] \\ &\subseteq \Pi_4 \text{TIME}[n^{\frac{c^6 \cdot (c-1)^2}{12}}], \end{aligned}$$

etc.

**Claim 2** The exponent  $e_k$  derived for  $\Sigma_k \text{TIME}[n] \subseteq \Pi_k \text{TIME}[n^{e_k}]$  is  $e_k = \frac{c^3 \cdot 2^{k-3} (c-1)^{2^{k-3}}}{k \cdot (3^{2^{k-4}} \cdot 4^{2^{k-5}} \cdot 5^{2^{k-6}} \dots (k-1))}$ .

**Proof.** By induction on  $k$ .  $\square$

We can simplify the expression to get

$$\begin{aligned} &\frac{c^3 \cdot 2^{k-3} (c-1)^{2^{k-3}}}{k \cdot (3^{2^{k-4}} \cdot 4^{2^{k-5}} \cdot 5^{2^{k-6}} \dots (k-1))} \\ &= \left( \frac{c^3 (c-1)}{k^{2-k+3} \cdot (3^{2^{-1}} \cdot 4^{2^{-2}} \cdot 5^{2^{-3}} \dots (k-1)^{2^{-k+3}})} \right)^{2^{k-3}}. \end{aligned}$$

Now,  $\frac{c^3 \cdot 2^{k-3} (c-1)^{2^{k-3}}}{k \cdot (3^{2^{k-4}} \cdot 4^{2^{k-5}} \cdot 5^{2^{k-6}} \dots (k-1))} < 1$  if and only if  $\frac{c^3 (c-1)}{k^{2-k+3} \cdot (3^{2^{-1}} \cdot 4^{2^{-2}} \cdot 5^{2^{-3}} \dots (k-1)^{2^{-k+3}})} < 1$ , so it suffices for us to analyze the latter expression.

The denominator numerically converges to  $3.81213 \dots$  as  $k \rightarrow \infty$ , so the calculation of  $c$  reduces to finding the positive root of  $c^3 \cdot (c-1) = 3.81213$ , or  $c \approx 1.7327 > \sqrt{3} + \frac{6}{10000}$ .

**Theorem 4**  $\text{NTIME}[n] \not\subseteq \text{DTISP}[n^{\sqrt{3}}, n^{o(1)}]$ .

A straightforward application of the methods of Tzourakis [18] further yields a lower bound on non-uniform machines for SAT (we employ the usual notation of  $\mathcal{C}/f(n)$  to denote class  $\mathcal{C}$  augmented with advice strings of length  $f(n)$  on inputs of length  $n$ ). We omit the details.

**Corollary 6**  $\text{NTIME}[n] \not\subseteq \text{DTISP}[n^{\sqrt{3}}, n^{o(1)}]/n^{o(1)}$ .

### 4.3 Solving tautologies with nondeterminism

A similar lower bound improvement can be derived for solving Boolean tautologies and related problems with non-deterministic time and small space. The best lower bound prior to ours was  $n^{1.324}$  time with  $n^{o(1)}$  space [7].

**Theorem 5** For every  $k \geq 2$  and rational  $c$  such that  $c < 1.337$ ,  $\text{coNTIME}[n] \not\subseteq \text{NTISP}[n^c, n^{o(1)}]$ .

As with Lemma 2, one can show that for  $k \geq 2$ ,  $\text{NTISP}[n^c, n^{o(1)}] \subseteq \Sigma_{2k-1} \text{TIME}[n^{c/k}]$ : two alternations at a time ( $\exists \forall$ ) are used to cut down the simulation time

of a block by a  $k$ th root, and a final existential alternation guesses the nondeterministic steps taken in the block. Hence if  $\text{coNTIME}[n] \subseteq \text{NTISP}[n^c, n^{o(1)}]$ , then

$$\begin{aligned} \Pi_3 \text{TIME}[n] &\subseteq \text{coNTIME}[n^{c^2}] \\ &\subseteq \text{NTISP}[n^{c^3}, n^{o(1)}] \subseteq \Sigma_3 \text{TIME}[n^{\frac{c^3}{2}}], \end{aligned}$$

$$\begin{aligned} \Pi_5 \text{TIME}[n] &\subseteq \text{NTISP}[n^{(\frac{c^3}{2})^2 \cdot c^3}, n^{o(1)}] \\ &\subseteq \Sigma_5 \text{TIME}[n^{\frac{c^9}{12}}], \end{aligned}$$

$$\begin{aligned} \Pi_7 \text{TIME}[n] &\subseteq \text{NTISP}[n^{(\frac{c^9}{12})^2 \cdot (\frac{c^3}{2})^2 \cdot c^3}, n^{o(1)}] \\ &\subseteq \Sigma_7 \text{TIME}[n^{\frac{c^{27}}{2304}}], \end{aligned}$$

*etc.*, and the expression for the exponent derived in the  $i$ th inclusion satisfies the recurrence  $e(1) = c^3/2$ ,  $e(i) = \frac{i}{i+1} \cdot e(i-1)^3$ . The bound on  $c$  such that for some  $k$  we have  $\Pi_k \text{TIME}[n] \subseteq \Sigma_k \text{TIME}[o(n)]$  converges to  $c \rightarrow 1.337 \dots$  as  $k$  increases.

#### 4.4 Better lower bounds for alternating time

The method also can be used to improve known lower bounds for alternating linear time. Here, we will just show the argument for  $\Sigma_2$ .

**Theorem 6**  $\Sigma_2 \text{TIME}[n] \not\subseteq \text{DTISP}[n^{f(k)^2}, n^{o(1)}]$ , for all  $k \geq 2$ . Note  $\lim_{k \rightarrow \infty} f(k)^2 = 2.7556 \dots$ .

**Proof.** From  $\Sigma_2 \text{TIME}[n] \subseteq \text{DTISP}[n^c, n^{o(1)}] \subseteq \Pi_2 \text{TIME}[n^{c/2}]$ , we derive

$$\begin{aligned} \Sigma_3 \text{TIME}[n] &\subseteq \Sigma_2 \text{TIME}[n^{c/2}] \\ &\subseteq \text{DTISP}[n^{c^2/2}, n^{o(1)}] \subseteq \Pi_3 \text{TIME}[n^{c^2/6}], \end{aligned}$$

$$\begin{aligned} \Sigma_4 \text{TIME}[n] &\subseteq \Sigma_2 \text{TIME}[n^{(c^2/6) \cdot c/2}] \\ &\subseteq \text{DTISP}[n^{c^4/12}, n^{o(1)}] \subseteq \Pi_4 \text{TIME}[n^{c^4/48}], \end{aligned}$$

*etc.*, and the bound on  $c$  that yields a contradiction converges to  $c \rightarrow 2.7556 \dots$  as  $k$  increases. More precisely, we get a contradiction when  $c < f(k)^2$ , for some  $k$ , where  $f$  is the function from Section 4.1.  $\square$

One can complete the columns of Table 1 for RAMs in similar fashion.

## 5 Improving time lower bounds for a strong brand of offline Turing machines

The above method makes it also possible to strengthen time lower bounds for a type of Turing machine that is a hybrid between a random access machine and a one-tape machine. The model has:

- an input tape that is read-only, random access,
- a small storage of  $n^{o(1)}$  bits that is read-write, random access, and
- an unbounded one-dimensional tape that is read-write with sequential (two-way) access.

It is important to observe that lower bounds on these machines are not as straightforward as one might think, *e.g.* these machines can recognize palindromes in linear time and logarithmic space.<sup>3</sup> Previously, an  $n^{\sqrt{3/2-\epsilon}} \approx n^{1.22}$  time lower bound for SAT was provable for this machine model, using a simulation due to Maass and Schorr [12] (independently re-discovered recently by Van Melkebeek and Raz [13]). An  $n^{\sqrt[4]{3/2-\epsilon}} \approx n^{1.1}$  bound proved by Kannan [10] in 1983 for a more restricted machine model can be easily seen to hold for the above as well. Our improvement pushes the lower bound to greater than  $n^{5/4}$ . Letting  $\text{DTIME}_1[t]$  denote the relevant time class for the aforementioned machine model, the lower bound is the following.

**Theorem 7**  $\text{NTIME}[n] \not\subseteq \text{DTIME}_1[n^{1.268}]$ .

**Proof.** Our main tool is another simulation lemma, found in the work of Van Melkebeek and Raz.

**Lemma 5** (VAN MELKEBEEK AND RAZ [13])

$\text{DTIME}_1[t] \subseteq \Sigma_2 \text{TIME}[t^{2/3+o(1)}]$ . In general,  $\text{DTIME}_1[t] \subseteq \Sigma_{k+1} \text{TIME}[t^{(k+1)/(2k+1)}]$  for all  $k \geq 1$ .

Now we prove the theorem: suppose  $\text{NTIME}(n) \subseteq \text{DTIME}_1[n^c]$ . Then  $\Pi_2 \text{TIME}[n] \subseteq \Pi_1 \text{TIME}[n^c] \subseteq \text{DTIME}_1(n^{c^2}) \subseteq \Sigma_2 \text{TIME}[n^{\frac{2}{3}c^2}]$ . Clearly there is a contradiction if  $c < \sqrt{\frac{3}{2}}$  (the previously known lower bound). If this is not the case, then as before we can use the derived inclusion  $\Pi_2 \text{TIME}[n] \subseteq \Sigma_2 \text{TIME}[n^{\frac{2}{3}c^2}]$ .

The induction hypothesis here is

<sup>3</sup>Contrast this machine model with the standard multi-tape Turing machine, where a  $O(\log n)$  space bound implies a  $\Omega(n^2)$  time lower bound for recognizing palindromes [4]. Santhanam [16] gave an efficient reduction from palindromes to SAT in this model, resulting in an  $\Omega(n^2/\text{polylog}(n))$  time lower bound for SAT on multi-tape machines.

For all  $i = 2, \dots, k-1$ ,  $\Pi_i \text{TIME}[n] \subseteq \Sigma_i \text{TIME}[n^{e(i)}]$ ,  
where  $e$  is defined inductively by  $e(0) = c$ ,  $e(k+1) =$   
 $\left(\frac{k+1}{2k+1}c^2\right) \prod_{i=1}^k e(i)$ . Thus, applying the lemma above,

$$\begin{aligned} \Pi_{k+1} \text{TIME}[n] &\subseteq \Pi_k \text{TIME}[n^{e(k+1)}] \\ &\subseteq \dots \subseteq \Pi_2 \text{TIME}[n^{\prod_{i=2}^k e(i)}] \\ &\subseteq \text{DTIME}_1[n^{c^2 \prod_{i=2}^k e(i)}] \\ &\subseteq \Sigma_{k+1} \text{TIME}[n^{\frac{k+1}{2k+1}c^2 \prod_{i=2}^k e(i)}], \end{aligned}$$

so

$$\Pi_{k+1} \text{TIME}[n] \subseteq \Sigma_{k+1} \text{TIME}[n^{e(k+1)}].$$

Simplifying as in previous cases, one can derive the recurrence for  $e$  to be  $e(1) = c$ ,  $e(k+1) = \frac{k+1}{k} \cdot \frac{2k-1}{2k+1} \cdot e(k)^2$ . Let  $c_k \in (1, 2)$  be such that  $e(k) = 1$  when  $c = c_k$ . As  $k \rightarrow \infty$ ,  $c_k > 1.2684$ , so the theorem follows.  $\square$

To complete the rest of Table 1 for this machine model, we simply observe that the same analysis holds, except that the ‘‘base case’’ for  $e$  changes. For example, the proof of the lower bound for  $\Sigma_2 \text{TIME}[n]$  results in the expression  $e(2) = \frac{2c}{3}$ ,  $e(k+1) = \frac{k+1}{k} \cdot \frac{2k-1}{2k+1} \cdot e(k)^2$ , yielding a  $n^{1.609}$  lower bound.

**Remark 2** *Van Melkebeek and Raz gave similar lower bounds for SAT on co-nondeterministic machines under the same model, as well as lower bounds when the sequential access tape is  $k$ -dimensional for constant  $k$ . Our method can be easily applied to improve these bounds as well.*

## 6 Lower bounds for bounded nondeterminism

In the remainder of the paper, we investigate how indirect diagonalization ideas using alternation can be further extended to prove lower bounds on *bounded* nondeterministic computation. Define  $\text{NTIBI}[t(n), b(n)]$  to be the class of languages recognized by  $t(n)$  *time* (the TI) random access Turing machines that use at most  $b(n)$  nondeterministic *bits* (the BI). More precisely, when given an input  $x$ , a characteristic machine for this class guesses  $b(|x|)$  bits on a special tape and then runs deterministically for  $t(|x|)$  time using the input tape, the special tape, and some number of worktapes.<sup>4</sup> We prove the separation:

**Theorem 8** *For all  $\varepsilon > 0$ , there is  $c_\varepsilon > 1$  such that  $\text{NTIBI}[n, n^\varepsilon] \not\subseteq \text{DTISP}[n^{c_\varepsilon}, n^{o(1)}]$ .*

<sup>4</sup>Note  $\text{NTIBI}[t, b] = \text{GC}[b, \text{DTIME}[t]]$ , where GC is the guess-and-check model of Cai and Chen [1]. We found the NTIBI notation more convenient for our purposes.

That is, even with only  $n^{1/1000}$  nondeterministic moves, there is still a non-linear time separation of nondeterminism from deterministic small space. This theorem has the interesting corollary that for all  $k$ , Boolean circuit satisfiability for circuits with  $n$  inputs and  $n^k$  gates and wires cannot be solved in  $n^{k+o(1)}$  time and small space.

In order to achieve a contradiction, we need a relationship between classes to contradict. The following hierarchy theorem for time and bits suffices.

**Theorem 9** *For any polynomial  $t(n) \geq n$ , and for all rational  $\varepsilon \in (0, 1)$ ,  $\gamma \in (0, 1)$ ,*

$$\text{coNTIBI}[t^{1+\gamma}, t^{\varepsilon+\gamma}] \not\subseteq \text{NTIBI}[t, t^\varepsilon].$$

**Proof.** Standard diagonalization. First, observe one can enumerate the set of random access Turing machines  $\{M_i\}$  using  $t(n)$  time and  $t^\varepsilon(n)$  nondeterministic bits in a standard way (perhaps the only difficulty in the proof is that  $t^\varepsilon(n)$  is computable in  $O(t(n))$  time). Define  $M'$  that on  $x$  determines  $M_x$ , universally guesses  $t^{\varepsilon+\gamma}(|x|)$  bits on a special tape, then simulates  $M_x(x)$  with the special tape, returning the opposite answer. It takes  $O(t^{\varepsilon+\gamma})$  time to write down the bits, and simulation of a deterministic time  $t$  machine can certainly be done in  $O(t^{1+\gamma})$  time. Since  $M'(x) = \neg M_x(x)$ , it is clear that if  $M_x$  is nondeterministic and  $M'$  is co-nondeterministic, then  $M_x(x)$  accepts iff  $M'(x)$  rejects.  $\square$

**Proof of Theorem 8.** Assume the opposite for contradiction, *i.e.* there is  $\varepsilon > 0$ , such that for all  $c > 1$ ,  $\text{NTIBI}[n, n^\varepsilon] \subseteq \text{DTISP}[n^c, n^{o(1)}]$ . Choose  $\varepsilon' \in (0, \min\{\varepsilon, c/2\})$  and  $t(n) \geq n^{1/\varepsilon'}$ .

By padding,  $\text{NTIBI}[t, t^\varepsilon] \subseteq \text{DTISP}[t^c, t^{o(1)}]$  for all  $c > 1$  and  $t(n) \geq n$ . Since DTISP is closed under complement, we have  $\text{coNTIBI}[t, t^\varepsilon] \subseteq \text{DTISP}[t^c, t^{o(1)}]$  as well. Now, a time  $t^c$  and space  $t^{o(1)}$  machine  $M$  can be simulated by a  $\Sigma_2$  machine of the kind

$$\begin{aligned} &(\exists \text{ configurations } C_1, \dots, C_{t^{\varepsilon'}} \text{ of } M \text{ on } x) \\ &(\forall i \in \{1, \dots, t^{c-\varepsilon'} + 1\}) \\ &[C_{i-1} \text{ leads to } C_i \text{ in } t^{c-\varepsilon'} \text{ time}], \end{aligned}$$

where  $C_0$  and  $C_{t^{\varepsilon'}+1}$  are the initial and accept configurations, respectively. The second and third lines describing the  $\Sigma_2$  computation above is a  $\text{coNTIBI}[N^{(c-\varepsilon')/\varepsilon'}, \log N]$  computation, where  $N = t^{\varepsilon'} + n \in O(t^{\varepsilon'})$ . Therefore the second and third lines can be simulated in  $\text{DTIME}[N^{c(c-\varepsilon')/\varepsilon'}]$  by assumption (and that  $c - \varepsilon' > \varepsilon'$ , since  $\varepsilon' < c/2$ ). Thus the above is equivalent to a computation of the form:

$$\left[ \begin{array}{l} (\exists \text{ configurations } C_1, \dots, C_{t^{\varepsilon'}} \text{ of } M \text{ on } x) \\ \text{a deterministic } t^{c(c-\varepsilon')} \text{ time computation on} \\ \langle x, C_1, \dots, C_{t^{\varepsilon'}} \rangle \end{array} \right],$$

which is an  $\text{NTIBI}[t^{c(c-\varepsilon')}, t^{\varepsilon'+o(1)}]$  computation.

Therefore we obtain

$$\text{coNTIBI}[t, t^\varepsilon] \subseteq \text{NTIBI}[t^{c(c-\varepsilon')}, t^{\varepsilon'+o(1)}].$$

If  $c(c-\varepsilon') < 1$ , then we have a contradiction with Theorem 9, as this implies  $\text{coNTIBI}[t, t^\varepsilon] \subseteq \text{NTIBI}[t^{1-\gamma}, t^{\varepsilon-\gamma}]$  for some  $\gamma > 0$ .

It remains for us to show that, given an  $\varepsilon' > 0$  that there is  $c > 1$  such that  $c(c-\varepsilon') < 1$ . Note  $d(d-\varepsilon') = 1$  has the unique positive solution  $d = (\varepsilon' + \sqrt{(\varepsilon')^2 + 4})/2$ . Therefore, for any  $\varepsilon' > 0$  we get  $d > 1$ , so any  $c \in (1, d)$  suffices.  $\square$

Notice in the above we needed exactly that  $c(c-\varepsilon') < 1$ ,  $\varepsilon' < \varepsilon$ , and  $\varepsilon' < c/2$ . So the maximum  $\varepsilon$  and  $c$  we could have are  $\varepsilon < 1/\sqrt{2}$  and  $c < \sqrt{2}$ . (To see this, note that  $c$  is maximized by making  $\varepsilon'$  as large as possible; change the inequalities to equations, and solve.) For some examples of concrete  $c > 1$ , we give the maximum  $\varepsilon > 0$  allowed by the above:

- $\text{NTIBI}[n, n^{\frac{1}{\sqrt{2}}-\varepsilon}] \not\subseteq \text{DTISP}[n^{1.414}, n^{o(1)}]$
- $\text{NTIBI}[n, n^{0.451}] \not\subseteq \text{DTISP}[n^{1.25}, n^{o(1)}]$
- $\text{NTIBI}[n, n^{0.0199}] \not\subseteq \text{DTISP}[n^{1.01}, n^{o(1)}]$

These separations have application to real satisfiability problems, as we now demonstrate.

**Lemma 6** Fix  $k \geq 1$ . If Boolean satisfiability on circuits with  $N$  inputs and  $N^k$  gates and wires is in  $\text{DTISP}[n^c, n^{o(1)}]$  for some  $c \geq 1$ , then  $\text{NTIBI}[n, n^{1/k}] \subseteq \text{DTISP}[n^c \text{poly}(\log n), n^{o(1)}]$ .

**Proof.** (Sketch) Pippenger and Fischer show that for deterministic multitape  $M$  running in time  $t$  there is a Boolean circuit  $C$  of  $O(t \log t)$  gates such that  $M(x, y)$  accepts iff  $C(x, y) = 1$ . Let  $C_x$  be  $C$  with the input bits of  $x$  hardcoded. Moreover,  $C_x$  can be constructed in  $O(t(\log t)^2)$  time and  $O(\log t)$  space (cf. Tourlakis [18]).

Suppose the problem of the lemma is solvable in  $n^c$  time and  $n^{o(1)}$  space. Take  $M'$  to be an arbitrary TM using  $n^\varepsilon$  nondeterministic bits running in  $O(n)$  time, so  $M'(x)$  accepts iff there is a  $y$  of length  $|x|^\varepsilon$  such that  $M(x, y)$  accepts, for some deterministic linear time  $M$ . Thus  $M'(x)$  accepts iff there is an input  $y$  such that  $C_x(y) = 1$ . Hence to simulate  $M'$  on  $x$ , we can reduce it to the circuit  $C_x$  of  $O(|x| \log |x|)$  gates with  $|x|^\varepsilon$  input size (note this is a logspace reduction), then solve the circuit satisfiability problem for  $C_x$  in  $O(|x|^c \text{poly}(\log(|x|)))$  time and  $|x|^{o(1)}$  space, by hypothesis.  $\square$

**Corollary 7** For all  $k \geq 1$ , there exists  $c_k > 1$  such that Boolean satisfiability on circuits with  $n$  inputs and  $n^k$  gates requires  $n^{k \cdot c_k}$  time on a deterministic random access machine using  $n^{o(1)}$  space.

## 7 Conclusion

We have demonstrated an inductive method for utilizing the polynomial hierarchy in proving lower bounds on concrete problems such as SAT. Our approach is extremely general, and can essentially be applied to any lower bound on  $\Sigma_k \text{TIME}$  where the class  $\mathcal{C}$  being shown as weak is sped up using a finite number of alternations. We also showed how existing lower bound arguments can yield lower bounds on bounded nondeterminism classes, due to the  $O(\log n)$ -bit universal quantifier inside a  $\Sigma_2 \text{TIME}$  simulation of  $\text{DTISP}$ .

Almost surely, we have not yet completely exploited the full “power of inductive thinking” in the results given here. In fact, we conjecture that a quadratic time lower bound for SAT on small-space RAMs is not only possible but also achievable via current techniques. The success of this inductive method raises compelling research questions. For example, given a class  $\mathcal{C}$  to prove lower bounds on, which class separations are most useful in an indirect diagonalization argument against  $\mathcal{C}$ ? The results of this paper suggest that, for many cases, the question’s answer is simply: *as many separations as possible*.

## 8 Acknowledgements

The author thanks his advisor Manuel Blum for his boundless encouragement and demand for clarity. Anupam Gupta, Richard Statman, Jon Kleinberg, Virginia Vassilevska, Maverick Woo, and an anonymous reviewer gave many helpful comments on both substance and style. This work owes a debt to Dieter van Melkebeek, whose commentary and enthusiasm inspired the author to push the SAT lower bound for small-space RAMs further towards  $n^2$ .

## References

- [1] L. Cai and J. Chen. On the Amount of Nondeterminism and the Power of Verifying. *SIAM J. Comput.* 26(3):733–750, 1997.
- [2] A. K. Chandra, D. Kozen, and L. J. Stockmeyer. Alternation. *Journal of the ACM* 28(1):114–133, 1981.
- [3] A. K. Chandra and L. J. Stockmeyer. Alternation. In *Proceedings of IEEE FOCS* 98–108, 1976.
- [4] A. Cobham. The recognition problem for the set of perfect squares. In *IEEE Symposium on Switching and Automata Theory*, 78–87, 1966.

- [5] S. A. Cook. Short Propositional Formulas Represent Nondeterministic Computations. *Inf. Process. Lett.* 26(5): 269-270 (1988)
- [6] L. Fortnow. Nondeterministic Polynomial Time Versus Nondeterministic Logarithmic Space: Time-Space Tradeoffs for Satisfiability. In *Proceedings of IEEE Conference on Computational Complexity*, 52–60, 1997.
- [7] L. Fortnow and D. v. Melkebeek. Time-Space Tradeoffs for Nondeterministic Computation. In *Proceedings of the IEEE Conference on Computational Complexity*, 2–13, 2000.
- [8] Y. Gurevich and S. Shelah. Nearly Linear Time. In *Logic at Botik'89, Symposium on Logical Foundations of Computer Science*, Springer LNCS 363:108–118, 1989.
- [9] R. J. Lipton and A. Viglas. On the Complexity of SAT. In *Proceedings of IEEE FOCS*, 459-4-64, 1999.
- [10] R. Kannan. Alternation and the power of nondeterminism. In *Proceedings of ACM STOC*, 344–346, 1983.
- [11] R. Kannan. Towards Separating Nondeterminism from Determinism. *Mathematical Systems Theory* 17(1):29–45, 1984.
- [12] W. Maass and A. Schorr. Speed-Up of Turing Machines with One Work Tape and a Two-Way Input Tape. *SIAM J. Comput.* 16(1):195–202, 1987.
- [13] D. van Melkebeek and R. Raz. A Time Lower Bound for Satisfiability. In *Proceedings of ICALP*, 971–982, 2004.
- [14] D. van Melkebeek. Time-Space Lower Bounds for NP-Complete Problems. In G. Paun, G. Rozenberg, and A. Salomaa (eds.), *Current Trends in Theoretical Computer Science*, 265–291, World Scientific, 2004.
- [15] V. Nepomnjaščii. Rudimentary predicates and Turing calculations. *Soviet Mathematics Doklady*, 11:1462-1465, 1970.
- [16] R. Santhanam. Lower bounds on the complexity of recognizing SAT by Turing machines. *Info. Proc. Lett.* 79(5):243–247, 2001.
- [17] C. P. Schnorr. Satisfiability is Quasilinear Complete in NQL. *Journal of the ACM* 25(1):136–145, 1978.
- [18] I. Tourlakis. Time-Space Lower Bounds for SAT on Uniform and Non-Uniform Machines. In *Proceedings of the IEEE Conference on Computational Complexity*, 22–33, 2000.

## 9 Appendix

**Proof of Lemma 2.** (Sketch) Fix a deterministic machine  $M$  using time  $t(n)$  and space  $s(n)$ . We first show how to get  $(ts^{k-1})^{1/k}$  time with  $2k$  quantifiers (a  $\Sigma_{2k}$  TIME $[(ts^{k-1})^{1/k}]$  machine) then show how the construction can be modified to get only  $\Sigma_k$  TIME $[(ts^{k-1})^{1/k}]$ . The key idea is to simulate the well-known proof that  $\text{SPACE}[s] \subseteq \text{ATIME}[s^2]$ , but to guess more configurations up front, reducing the number of alternations overall. A machine  $N$  with  $2k$  alternations is as follows.

$N(x)$ : Let  $C_0$  and  $C_{t+1}$  be the unique initial and accept configurations of  $M(x)$ . Return **Simulate** $(C_0, C_{t+1}, k)$ .

**Simulate** $(C_i, C_j, j)$ :

If  $j = 0$  then *accept* iff  $C_i$  leads to  $C_j$  in at most  $(ts^{k-1})^{1/k}$  steps.

Else:

- *Existentially* guess configurations  $C_1^j, \dots, C_{(t/s)^{1/k}}^j$  of  $M(x)$ . If  $C_1^j \neq C_i$  then *reject*. If  $C_{(t/s)^{1/k}}^j \neq C_j$  then *reject*.
- *Universally* choose  $i_j \in \{1, \dots, (t/s)^{1/k}\}$ , and **Simulate** $(C_{i_j}^j, C_{i_j+1}^j, j - 1)$ .

It is straightforward to verify that this procedure works just like Savitch's theorem, except we are guessing more "midpoint" configurations before each recursive call.

**Simulate** with  $j := k$  clearly has  $2k$  alternations, guessing  $O((t/s)^{1/k} \cdot s) = O((ts^{k-1})^{1/k})$  bits existentially and  $O(\log t)$  bits universally between each recursive call, and running for  $O((ts^{k-1})^{1/k})$  deterministic time in the base case (on a RAM). Thus the procedure takes  $O((ts^{k-1})^{1/k})$  time overall.

How do we reduce the number of alternations from  $2k$  to  $k$ ? We can exploit the fact that the computation is deterministic. We could just as easily rewrite **Simulate** to be:

**Simulate2** $(C_i, C_j, j)$ :

If  $j = 0$  then *accept* iff  $C_i$  leads to  $C_j$  in at most  $(ts^{k-1})^{1/k}$  steps.

Else:

- *Universally* choose configurations  $C_1^j, \dots, C_{(t/s)^{1/k}}^j$  of  $M(x)$ . If  $C_1^j \neq C_i$  then *accept*. If  $C_{(t/s)^{1/k}}^j = C_j$  then *accept*.
- *Existentially* choose  $i_j \in \{1, \dots, (t/s)^{1/k}\}$ .
- $\neg$ **Simulate2** $(C_{i_j}^j, C_{i_j+1}^j, j - 1)$ .

To verify that  $C_i$  leads to  $C_j$ , we consider all sequences of configurations where the first configuration is  $C_i$ , but the last one is *not*  $C_j$ . We verify that for any such sequence, there are two adjacent configurations that do *not* lead from one to the other. Since all sequences from  $C_i$  to some  $C'_j \neq C_j$  fail to work, it must be that  $C_i$  leads to  $C_j$ . Clearly, **Simulate2** runs in the same time bound as **Simulate**, but the quantifiers start with a  $\forall$  instead.

Our final solution is to make the two procedures mutually recursive: rewrite **Simulate** so that it calls **Simulate2**, and vice-versa. Then, the number of alternations in  $N(x)$  becomes exactly  $k$ .  $\square$