Alternation-Trading Proofs, Linear Programming, and Lower Bounds

RYAN WILLIAMS, Stanford University

A fertile area of recent research has demonstrated concrete polynomial time lower bounds for natural hard problems on restricted computational models. Among these problems are Satisfiability, Vertex Cover, Hamilton Path, MOD$_6$-SAT, Majority-of-Majority-SAT, and Tautologies, to name a few. The proofs of these lower bounds follow a proof-by-contradiction strategy that we call *resource-trading* or *alternation-trading*. An important open problem is to determine how powerful such proofs can possibly be.

We propose a methodology for studying these proofs that makes them amenable to both formal analysis and automated theorem proving. We prove that the search for better lower bounds can often be turned into a problem of solving a large series of linear programming instances. Implementing a small-scale theorem prover based on these results, we extract new human-readable time lower bounds for several problems and identify patterns that allow for further generalization. The framework can also be used to prove concrete limitations on the current techniques.

Categories and Subject Descriptors: F.1.3 [**Theory of Computation**]: Complexity Measures and Classes

General Terms: Algorithms, Theory

Additional Key Words and Phrases: alternation, time-space tradeoffs, lower bounds, linear programming

ACM Reference Format:
Ryan Williams, 2013. Alternation-trading proofs, linear programming, and lower bounds. *ACM Trans. Comput. Theory* V, N, Article A (January YYYY), 40 pages.
DOI:http://dx.doi.org/10.1145/0000000.0000000

1. INTRODUCTION

This work is concerned with proving new limitations on computers by exploiting their capabilities. Many known lower bounds for natural problems use a type of algorithmic argument that we call a *resource-trading proof*. Informally, such a proof uses four basic steps:

(1) Assume a hard problem Π can be solved in n^c time with resources R. (Let's abbreviate the class of such problems as $R[n^c]$.) We wish to obtain a contradiction. For example, $R[n^c]$ may be $\mathsf{DTISP}[n^c, \text{poly}(\log n)]$, the set of problems solvable in n^c time and $\text{poly}(\log n)$ space (simultaneously), and Π may be the SAT problem.

(2) Prove a *Speedup Lemma* that "trades time for resources". Informally, such a lemma proves that the class $R[t]$ is contained in a class $S[o(t)]$, for a more powerful resource S and time bound t. For example, $S[t]$ may be the class of problems solvable by alternating machines in time t. Nepomnjascii [Nepomnjascii 1970] showed that every $\text{poly}(\log n)$ space algorithm running in n^k time can be simulated by a Σ_k machine (using k alternations) that runs in only $O(n \cdot \text{poly}(\log n))$ time.

(3) Prove a *Slowdown Lemma* that "trades resources for time", where $S[t]$ is shown to be in $R[t^d]$, for a small constant $d \geq 1$. This typically uses the assumption that $\Pi \in R[n^c]$. For example,

This material is based on work supported in part by NSF grant CCR-0122581 at CMU, and NSF grant CCF-0832797 at Princeton University/IAS. Any opinions, findings and conclusions or recommendations expressed are those of the author. Email: `rrw@cs.stanford.edu`.

DOI:http://dx.doi.org/10.1145/0000000.0000000

if SAT has an n^c time, $\mathrm{poly}(\log n)$ space algorithm, then (by a strong form of the Cook-Levin theorem) it follows that every language accepted by a nondeterministic time-t algorithm can be perfectly simulated by a deterministic algorithm running in $t^{c+o(1)}$ time and $\mathrm{poly}(\log t)$ space, for all polynomials $t(n)$. Consequently, every Σ_k machine running in t time has an equivalent deterministic algorithm running in $t^{c^k+o(1)}$ time and $\mathrm{poly}(\log t)$ space.

(4) Combine (2) and (3) to show $\mathscr{C}[t] \subseteq \mathscr{C}[t^{1-\varepsilon}]$, for some $\varepsilon > 0$ and complexity class $\mathscr{C}[t]$, implying a contradiction with a hierarchy theorem for $\mathscr{C}$. For example, if SAT has an n^c time, $\mathrm{poly}(\log n)$ space algorithm, then $\Sigma_2\mathsf{TIME}[t] \subseteq \mathsf{DTISP}[t^{c^2+o(1)}, \mathrm{poly}(\log t)] \subseteq \Pi_2\mathsf{TIME}[t^{c^2/2+o(1)}]$, where the first inclusion holds by (3) and the second holds by (2). If $c^2 < 2$, then we have derived $\Sigma_2\mathsf{TIME}[t] \subseteq \Pi_2\mathsf{TIME}[o(t)]$, which contradicts the known time hierarchy for alternating machines. This argument is precisely the $n^{\sqrt{2}-o(1)}$ lower bound of Lipton and Viglas [Lipton and Viglas 1999].

A detailed description of this style of argument can be found in Section 3. The basic strategy has been applied in many lower bound settings, dating back to the 70's. A partial list includes:

Time Versus Space. Hopcroft, Paul, and Valiant [Hopcroft et al. 1977] proved that $\mathsf{SPACE}[n] \not\subseteq \mathsf{DTIME}[o(n\log n)]$ for multitape Turing machines, by proving the "speedup lemma" that $\mathsf{DTIME}[t] \subseteq \mathsf{SPACE}[t/\log t]$ and invoking diagonalization. Their result was also extended to more general models [Paul and Reischuk 1981; Halpern et al. 1986].

Determinism vs Nondeterminism for Multitape Turing Machines. A celebrated result of Paul-Pippenger-Szemeredi-Trotter [Paul et al. 1983] is that $\mathsf{NTIME}[n] \neq \mathsf{DTIME}[n]$ for multitape Turing machines. The key component in the proof is the "speedup lemma" that $\mathsf{DTIME}[t] \subseteq \Sigma_4\mathsf{TIME}[t/\log^* t]$ for multitape TMs.

Small-Bounded Algorithms. In this setting, the computational model is a random access machine using small workspace ($n/(\log n)^c$, $n^{1-\varepsilon}$, and $n^{o(1)}$ are typical values). Non-trivial time lower bounds have been proved for NP-complete problems and problems higher in the polynomial hierarchy [Kanan 1984; Fortnow 1997; Lipton and Viglas 1999; Fortnow and van Melkebeek 2000; Fortnow et al. 2005; Williams 2006; Williams 2008]. The best known time lower bound for solving SAT with $n^{o(1)}$ space algorithms is $n^{2\cos(\pi/7)-o(1)} \geq n^{1.801}$ [Williams 2008]. This bound also holds for the MODm-SAT problem, for any composite m that is not a prime power. For nondeterministic algorithms using $n^{o(1)}$ space, the best known time lower bound for solving the coNP complete TAUTOLOGY problem was $n^{\sqrt{2}-o(1)}$ [Fortnow and van Melkebeek 2000].

Probabilistic and Quantum Space-Bounded Algorithms. Allender *et al.* [Allender et al. 2001] showed that Maj-Maj-SAT requires $n^{1+\Omega(1)}$ time to solve on unbounded error machines that use $n^{1-\varepsilon}$ space, for $\varepsilon > 0$. Diehl and Van Melkebeek [Diehl and van Melkebeek 2006] proved that for $k \geq 2$, k-QBF requires $\Omega(n^{k-o(1)})$ time with randomized two-sided error algorithms using $n^{o(1)}$ space. Viola [Viola 2007] has shown that 3-QBF requires $n^{1+\delta}$ time on Turing machines with a random access input tape and two-way read-write access to a random bit tape, for some $\delta > 0$. Van Melkebeek and Watson [van Melkebeek and Watson 2007] have shown time lower bounds for Maj-Maj-SAT with quantum algorithms.

General Multidimensional Turing Machines. This model has read-only random access to its input, an $n^{o(1)}$ read-write store, and read-write access to a d-dimensional tape for a fixed $d \geq 1$. This model generalizes several others, and is the most powerful (and physically realistic) model known where we still know non-trivial time lower bounds for SAT. Multidimensional TMs have been studied for many years; for instance, [Loui 1980; Paul and Reischuk 1981; Grigor'ev 1982; Kannan 1983; Maass and Schorr 1987; Liskiewicz and Lorys 1990; Dietzfelbinger and Hühne 1999; van

Melkebeek 2005; Williams 2006] proved lower bounds in this model, and the previous best bound for SAT was essentially $\Omega(n^{\sqrt{(d+2)/(d+1)}})$ time in the d-dimensional case.

Resource-trading proofs have been traditionally *ad hoc* in their design, making it hard to build intuition about them. Each argument has its own "custom-built" speedup and slowdown lemma, and in each setting, these lemmas are applied in rather non-trivial ways. One gets a sense that the space of all possible proofs might be difficult to systematically study.

1.1. Main Results

We introduce a methodology for reasoning about resource-trading proofs that is also practically implementable for finding short proofs of time lower bounds. We argue that for almost all known resource-trading lower bounds, the proofs can be reformulated in a way that the search for new lower bounds becomes a feasible problem that computers can help attack.[1] Informally, we show that the "hard work" in these proofs can be replaced by a series of linear programming problems. This approach not only aids us practically in the search for new lower bounds, but also allows us to understand the limits of what can be proved with slowdown and speedup lemmas.

We apply our methodology to several settings in which lower bounds have been previously obtained. In all cases, the resource being "traded" is alternations, so for the purposes of this work we call the proofs *alternation-trading*. We formalize the components used in prior work and their relevant properties, with the following results.

Deterministic Time-Space Lower Bounds for SAT and Beyond. Aided by results of a computer program, we show that any SAT algorithm running in $t(n)$ time and $s(n)$ space must satisfy $t(n) \cdot s(n) \geq n^{\phi - o(1)}$ infinitely often, where $\phi \approx 1.618\cdots$ is the golden ratio, and for specific choices of $t(n)$ and $s(n)$, we can prove better lower bounds. Previously, the best known result was $t(n) \cdot s(n) \geq n^{1.573}$ [Fortnow et al. 2005]. It has been conjectured that the current framework of alternation-trading proofs sufficed to prove an $n^{2-o(1)}$ time lower bound for SAT, against algorithms using $n^{o(1)}$ space.[2] We prove that it will be impossible to obtain n^2 with alternation-trading proofs, formalizing a folklore conjecture. A computer search over proofs of small length strongly suggests that the best known $n^{2\cos(\pi/7)-o(1)}$ lower bound [Williams 2008] is already optimal for the framework. Recent work of the author with Buss [Buss and Williams 2012] has in fact confirmed this conjecture, showing that the $2\cos(\pi/7)$ exponent is the best possible for alternation-trading proofs. We also prove lower bounds on the problem of determining validity of quantified Boolean formulas with at most k quantifier blocks (also known as QBF$_k$), showing that QBF$_k$ requires $\Omega(n^{k+1-\delta_k})$ time for $n^{o(1)}$ space algorithms, where $\delta_k < 0.2$ and $\lim_{k\to\infty} \delta_k = 0$.

Nondeterministic Time-Space Lower Bounds for Tautologies. Adapting our approach to this problem, a computer program found a very short proof improving upon Fortnow and Van Melkebeek's lower bound. Longer proofs suggested an interesting pattern. Joint work with Diehl and Van Melkebeek on these observations resulted in an $n^{4^{1/3}-o(1)} \geq n^{1.587}$ time lower bound [Diehl et al. 2011].Computer search suggests that the $4^{1/3}$ exponent is best possible for alternation-trading proofs. We can prove that it is not possible to obtain an n^{ϕ} time lower bound, where $\phi = 1.618\ldots$ is the golden ratio. This is somewhat surprising, since we have known for some time [Fortnow and van Melkebeek 2000] that an n^{ϕ} lower bound *is* provable for *deterministic* algorithms.

Lower Bounds for Multidimensional Turing Machines. Here, the method uncovers particular behavior in the best lower bound proofs, regardless of the dimension of the tapes. Studying computer

[1]We note that combinatorial arguments such as Santhanam's time-space lower bound for SAT on multitape Turing machines [Santhanam 2001] do not fall under the alternation-trading paradigm, but they are already known to have different limitations.

[2]The author could not find an explicit reference for this conjecture, but he has received several referee reports in the past that state it. Also see Lipton-Viglas [Lipton and Viglas 1999] in FOCS'99.

search results, an $\Omega(n^{r_d})$ time lower bound is formally proved for the d-dimensional case, where $r_d \geq 1$ is the root of a particular quintic $p_d(x)$ with coefficients depending on d. For example, $r_1 \approx 1.3009$, $r_2 \approx 1.1887$, and $r_3 \approx 1.1372$. Again, computer search suggests this is best possible, and we prove that it is impossible to push the time lower bound for d-dimensional TMs to $n^{1+1/(d+1)}$ with the current tools we have.

These limitations hold for other NP and coNP-hard problems as well; the only property required is that every set in $\mathsf{NTIME}[n]$ (respectively, $\mathsf{coNTIME}[n]$) has sufficiently efficient reductions to the problem. Furthermore, this approach is not limited to the above, and can be applied to the league of lower bounds discussed in Van Melkebeek's surveys [van Melkebeek 2004; van Melkebeek 2006]. We have chosen to present the above cases because in our opinion they are among the most interesting, and the results already illustrate a diversity of structure in alternation-trading proofs. This work promotes a new methodology for proving lower bounds, where prospective lower-bounders formalize their proof rules, write a program to test ideas and generate short proofs, then study the results and extrapolate new results.

1.2. Brief Intuition

Intuitively, the key insight behind our formulation is to separate the *discrete choices* in an alternation-trading proof from the *real-valued choices*, showing that the number of discrete choices can be made somewhat small, and the remaining real-valued choices can be posed as efficiently solvable subproblems. The discrete choices consist of the sequence of lemmas to apply in each step, and which complexity class $\mathscr{C}[t]$ to use in the proof by contradiction. We show how to reduce all alternation-trading proofs into a *normal form* which greatly reduces the number of necessary discrete choices, without loss of generality. Real-valued choices come from selecting t, as well as parameters arising from rule applications. We prove that once the discrete choices are made, the remaining real-valued problem can be expressed as an instance of linear programming. This makes it possible to search for new proofs via computer, and it provides an efficient mechanism for reasoning rigorously about alternation-trading proofs.

One cannot easily search over all possible proofs, as the number of discrete choices is still $\sim 2^n/n^{3/2}$ for proofs of n lines (proportional to the nth Catalan number). Nevertheless it is still feasible to search over all 20+ line proofs. These searches reveal regular patterns, indicating that certain strategies will be most successful in proving lower bounds; in each case we study, the resulting strategies are different. Following the strategies, we establish new lower bound proofs. The patterns also suggest how to prove limitations on the proof system.

1.3. Outline of the Paper

The rest of the paper is structured as follows. Section 2 and 3 present some necessary background on the alternation-trading paradigm. Section 4 focuses on time-space lower bounds for SAT: we formalize alternationg-trading proofs in this setting (Section 4.1), show how to reduce the optimal setting of parameters to an instance of linear programming (Section 4.2), present the results of our theorem-proving program, highlighting the best short lower bound proofs (Section 4.3), prove limitations on achieving large lower bounds within the framework (Section 4.4), and prove new time-space tradeoff lower bounds based on the empirical results (Section 4.5). A similar pattern is followed in Sections 5, 6, and 7, but for time-space lower bounds on quantified Boolean formulas with a fixed number of alternations, nondeterministic time-space lower bounds for Tautologies, and deterministic time lower bounds for multidimensional Turing machines, respectively. Section 8 concludes the paper.

2. PRELIMINARIES

We assume basic familiarity with Complexity Theory, especially with the concept of alternation [Chandra et al. 1981]. We use big-Ω notation in the infinitely often sense, so statements like "SAT is not in $O(n^c)$ time" are equivalent to "SAT requires $\Omega(n^c)$ time." All functions are assumed constructible within the appropriate bounds. Our default computational model is the random access machine (RAM), broadly construed: particular variants do not affect the results. In Section 7, we also consider a general *d-dimensional Turing machine* model, which has an input tape that is read-only with random access, a small storage of $n^{o(1)}$ bits that is read-write with random access, and an unbounded d-dimensional tape that is read-write with sequential (two-way) access.

$\mathsf{DTISP}[t(n), s(n)]$ is the class of languages accepted by a RAM running in $t(n)$ time and $s(n)$ space, simultaneously. For convenience, we set $\mathsf{DTS}[t(n)] := \mathsf{DTISP}[t(n)^{1+o(1)}, n^{o(1)}]$ to omit negligible $o(1)$ factors.

In order to properly formalize alternation-trading proofs, we introduce notation for alternating complexity classes which include *input constraints* between alternations. These constraints are critical for the formalism. (Several lower bounds rely on the fact that these input lengths can be small in certain interesting cases.)

Let us start with an example of the notation, then give a general definition. Let $a > 0$ and $b, c \geq 1$. Define

$$(\exists\ n^c)^b \mathsf{DTS}[n^a]$$

to be the class of languages recognized by a machine which, on an input x of length n, writes an $n^{c+o(1)}$ bit string y nondeterministically, takes at most $n^{b+o(1)}$ time to copy $n^{b+o(1)}$ bits z from the pair $\langle x, y\rangle$, then feeds z as input to a machine M running in $n^{a+o(1)}$ time and $n^{o(1)}$ space. Note the runtime of M is measured with respect to the initial input length n, not the latter input length $n^{b+o(1)}$ of z.

We generalize this definition as follows. For $i = 1, \ldots, k$, let $Q_i \in \{\exists, \forall\}$ and $a_i > 0$, $b_i \geq 1$. Define

$$(Q_1\ n^{a_1})^{b_2} (Q_2\ n^{a_2}) \cdots^{b_k} (Q_k\ n^{a_k})^{b_{k+1}} \mathsf{DTISP}[n^c, n^e]$$

to be the class of languages recognized by a machine M that, on input x of length n, has the following general behavior on input x:

```
Set $z_0 := x$.
For $i = 1, \ldots, k$,
    If $Q_i = \exists$, switch to *existential mode*.
    If $Q_i = \forall$, switch to *universal mode*.
    Guess an $n^{a_i+o(1)}$ bit string $y$ (universally or existentially).
    Using at most $n^{b_{i+1}+o(1)}$ time, copy at most $n^{b_{i+1}+o(1)}$ bits $z_i$ from the pair $\langle z_{i-1}, y\rangle$.
End for
Run an $n^{c+o(1)}$ time, $n^{e+o(1)}$ space machine on the input $z_k$, and output its decision.
```

When an input constraint b_i is unspecified, its default value is $\max\{a_i, 1\}$. We say that the existential and universal modes of an alternating computation are *quantifier blocks*, to reflect the complexity class notation. It is crucial to observe that the time bound in the ith quantifier block is measured with respect to n, the input to the *first quantifier block*.

Notice that by simple properties of nondeterminism and conondeterminism, we can combine adjacent quantifier blocks that are of the same type, e.g., $(\exists\ n^a)^a (\exists\ n^b)^b \mathsf{DTS}[n^c] = (\exists\ n^{\max\{a,b\}})^b \mathsf{DTS}[n^c]$. This useful property is exploited in alternation-trading proofs.

3. A SHORT INTRODUCTION TO ALTERNATION-TRADING PROOFS

In this section, we give a brief overview of the tools that have been used in prior work to prove time-space tradeoff lower bounds within the alternation-trading paradigm. We focus on deterministic time lower bounds for satisfiability for algorithms using $n^{o(1)}$ space, as this setting has received the most attention and the other relevant lower bound problems use analogous tools.

It is known that satisfiability of Boolean formulas in conjunctive normal form (SAT) is a complete problem under tight reductions for a somewhat small nondeterministic time complexity class. The class *Nondeterministic Quasi-linear Time* is defined as

$$\mathsf{NQL} := \bigcup_{c \geq 0} \mathsf{NTIME}[n \cdot (\log n)^c] = \mathsf{NTIME}[n \cdot poly(\log n)].$$

THEOREM 3.1 ([COOK 1988; SCHNORR 1978; TOURLAKIS 2001; FORTNOW ET AL. 2005]). *SAT is* NQL*-complete, under reductions in quasi-linear time and* $O(\log n)$ *space simultaneously, in the random access machine model. Moreover, each bit of the reduction can be computed in* $O(poly(\log n))$ *time and* $O(\log n)$ *space.*

This theorem has the following consequences. Let $\mathscr{C}[t(n)]$ represent a time $t(n)$ complexity class under one of the three computational models:

— deterministic RAM using time t and $t^{o(1)}$ space,
— co-nondeterministic RAM using time t and $t^{o(1)}$ space,
— d-dimensional Turing machine using time t (as defined in the Preliminaries).

Theorem 3.1 implies that if $\mathsf{NTIME}[n] \not\subseteq \mathscr{C}[t]$, then SAT $\notin \mathscr{C}[t]$, modulo polylogarithmic factors.

COROLLARY 3.2. *If* $\mathsf{NTIME}[n] \not\subseteq \mathscr{C}[t(n)]$*, then there is a* $k > 0$ *such that SAT* $\notin \mathscr{C}[t(n)/(\log t(n))^k]$.

Hence we want to prove $\mathsf{NTIME}[n] \not\subseteq \mathscr{C}[n^c]$ for a large constant $c > 1$. For the purposes of time lower bounds for small space algorithms, we work with $\mathscr{C}[n^c] = \mathsf{DTS}[n^c] = \mathsf{DTISP}[n^{c+o(1)}, n^{o(1)}]$. Van Melkebeek and Raz [van Melkebeek 2005] observed that a similar corollary holds for any problem Π such that SAT reduces to Π under logtime-computable quasi-linear reductions (*e.g.*, VERTEX COVER, HAMILTON PATH, 3-SAT, and MAX-2-SAT). It follows that, given a time lower bound for $\mathsf{NTIME}[n]$, similars time lower bounds hold for these specific problems as well.

Speedups, Slowdowns, and Contradictions. Now that our goal is to prove $\mathsf{NTIME}[n] \not\subseteq \mathsf{DTS}[n^c]$, how can we prove such a separation? In an alternation-trading proof, we assume that $\mathsf{NTIME}[n] \subseteq \mathsf{DTS}[n^c]$ and attempt to establish a contradiction, by applying two lemmas in such a way that a time hierarchy is violated. One lemma (called the "speedup lemma") takes a $\mathsf{DTS}[t]$ class and places it in an alternating class with runtime $o(t)$; the other (called the "slowdown lemma") takes an alternating class with runtime t and places it in a class with one less alternation and runtime approximately $O(t^c)$.

LEMMA 3.3 (SPEEDUP LEMMA). *Let* a, e, x *satisfy* $a \geq 1$, $0 \leq e \leq a - x$, *and* $0 \leq x \leq a - 1$. *Then (in the notation of Section 2)*

$$\mathsf{DTISP}[n^a, n^e] \subseteq (Q_1\ n^{x+e})^{\max\{1,x+e\}} (Q_2\ \log n)^{\max\{1,e\}} \mathsf{DTISP}[n^{a-x}, n^e],$$

for $Q_i \in \{\exists, \forall\}$ *where* $Q_1 \neq Q_2$. *In particular,*

$$\mathsf{DTS}[n^a] \subseteq (Q_1\ n^x)^{\max\{1,x\}} (Q_2\ \log n)^1 \mathsf{DTS}[n^{a-x}].$$

The Speedup Lemma dates back to work of Nepomnjascii [Nepomnjascii 1970] and Kannan [Kanan 1984]. We include its proof for completeness.

PROOF. Let M be a random access machine using n^a time and n^e space. To get a simulation of M of type

$$(\exists\ n^{x+e})^{\max\{1,x+e\}} (\forall\ \log n)^{\max\{1,e\}} \mathsf{DTISP}[n^{a-x}, n^e],$$

the simulation N on an input x existentially guesses a sequence of configurations $C_1, \ldots, C_{n^x}$ of $M(x)$. Next, N appends the initial configuration of $M(x)$ to the beginning of the sequence and the (WLOG, unique) accepting configuration to the end of the sequence. Then N universally guesses an $i \in \{0, \ldots, n^x\}$, and copies C_i and C_{i+1} over to a special register z, in $O(n^e)$ time. Finally N simulates

M using only x and z as input; it simulates $M(x)$ starting from configuration C_i and accepts if and only if C_{i+1} is reached within n^{a-x} steps.

It is easy to see that the simulation is correct. Note that the input constraints on the quantifier blocks are satisfied: after the first quantifier block, the input kept has length $n + n^{x+e+o(1)}$, and after the universal guess of i, the input kept is only x, C_i, and C_{i+1}, which together have size $n + 2n^e \leq n^{\max\{1,e\}+o(1)}$. □

Note that in the above alternating simulation, the input to the final DTISP computation is linear in $n + n^e$, *regardless of the choice of x*. This is a subtle property that is exploited heavily in alternation-trading proofs. The Slowdown Lemma is the following folklore result:

LEMMA 3.4 (SLOWDOWN LEMMA). *Let $a \geq 1$, $e \geq 0$, $a' \geq 0$, and $b \geq 1$. If $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^e]$, then for both $Q \in \{\exists, \forall\}$,*

$$(Q\ n^{a'})^b \mathsf{DTIME}[n^a] \subseteq \mathsf{DTISP}[n^{c\cdot\max\{a,a',b\}}, n^{e\cdot\max\{a,a',b\}}].$$

In particular, if $\mathsf{NTIME}[n] \subseteq \mathsf{DTS}[n^c]$, then

$$(Q\ n^{a'})^b \mathsf{DTIME}[n^a] \subseteq \mathsf{DTS}[n^{c\cdot\max\{a,a',b\}}].$$

PROOF. Let L be a language in the class $(Q\ n^{a'})^b \mathsf{DTIME}[n^a]$, and let A be a RAM of the appropriate type that recognizes L. On an input x of length n, A guesses a string y of length $n^{a'+o(1)}$, then feeds an $n^{b+o(1)}$ bit string z to $A'(z)$, where A' is a deterministic algorithm that runs in $O(n^a)$ time. Since $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^c, n^e]$ and DTISP is closed under complement, by padding we have

$$\mathsf{NTIME}[p(n)] \cup \mathsf{coNTIME}[p(n)] \subseteq \mathsf{DTISP}[p(n)^c, p(n)^e]$$

for all polynomials $p(n) \geq n$. Therefore A can be simulated with a deterministic algorithm B. As the total runtime of A is $n^{a'+o(1)} + n^{b+o(1)} + O(n^a)$, B runs in $n^{c\cdot\max\{a,a',b\}+o(1)}$ time and $n^{e\cdot\max\{a,a',b\}+o(1)}$ space. □

The final component of an alternation-trading proof is a time hierarchy theorem, the most general of which is the following, provable by a simple diagonalization argument.

THEOREM 3.5 (ALTERNATING TIME HIERARCHY). *For $k \geq 0$, for all $Q_i \in \{\exists, \forall\}$, $a_i > a'_i \geq 1$, and $b_i \geq b'_i \geq 1$,*

$$(Q_1\ n^{a_1})^{b_2} \cdots^{b_k} (Q_k\ n^{a_k})^{b_{k+1}} \mathsf{DTS}[n^{a_{k+1}}] \nsubseteq (R_1\ n^{a'_1})^{b'_2} \cdots^{b'_k} (R_k\ n^{a'_k})^{b'_{k+1}} \mathsf{DTS}[n^{a'_{k+1}}],$$

where $R_i \in \{\exists, \forall\}$ and $R_i \neq Q_i$.

Remark 3.6. Are alternation-trading proofs subject to the usual complexity barriers?[3] The Slowdown Lemma relativizes, but the Speedup Lemma does not relativize in most oracle models, for the simple reason that the original (simulated) machine runs for longer time than the host (simulating) machine, and can therefore ask longer queries. This is typically the case. For example, the proof that $\mathsf{NTIME}[n] \neq \mathsf{DTIME}[n]$ is non-relativizing, because a powerful enough oracle can render the two classes equal. Moreover, alternation-trading proofs do not seem to fit in the "natural proofs" framework either. Natural proofs require that the underlying proof can be used to test whether a given Boolean function is hard, and that test must give a "hard" verdict on a significant fraction of the possible functions. Alternation-trading proofs do not fit this mold: they are only concerned with proving lower bounds for *complete* problems of a very specific kind. Therefore, we consider alternation-trading proofs to be in that rare class of non-relativizing and non-naturalizing lower bounds (but acknowledge that our belief is not unanimously held).

[3]Background on the known barriers in complexity theory can be found in Arora and Barak ([Arora and Barak 2009], Chapter 23).

Two Instructive Examples. In order to understand alternation-trading proofs, it is necessary to consider some examples. The art behind their construction consists of finding the proper sequence of rules to apply, and the right settings of the parameter x in the Speedup Lemma.

(1) In FOCS'99, Lipton and Viglas proved that SAT cannot be solved by algorithms running in $n^{\sqrt{2}-\varepsilon}$ time and $n^{o(1)}$ space, for all $\varepsilon > 0$. Their proof can be summarized as follows: by Theorem 3.1, the assumption that there is such a SAT algorithm implies that $\mathsf{NTIME}[n] \subseteq \mathsf{DTS}[n^c]$ with $c^2 < 2$. Then

$$\begin{aligned}(\exists\, n^{2/c^2})(\forall\, n^{2/c^2})\mathsf{DTS}[n^{2/c^2}] &\subseteq (\exists\, n^{2/c^2})\mathsf{DTS}[n^{2/c}] \quad \text{(Slowdown Lemma)}\\ &\subseteq \mathsf{DTS}[n^2] \quad \text{(Slowdown Lemma)}\\ &\subseteq (\forall\, n)(\exists\, \log n)\mathsf{DTS}[n] \quad \text{(Speedup Lemma, with } x = 1).\end{aligned}$$

But $(\exists\, n^{2/c^2})(\forall\, n^{2/c^2})\mathsf{DTS}[n^{2/c^2}] \subseteq (\forall\, n)(\exists\, \log n)\mathsf{DTS}[n]$ contradicts Theorem 3.5. In fact, one can show that if $c^2 = 2$, we still have a contradiction with $\mathsf{NTIME}[n] \subseteq \mathsf{DTS}[n^c]$, so the ε can be removed from the previous statement and state that SAT cannot be solved in $n^{\sqrt{2}}$ time and $n^{o(1)}$ exactly.[4]

(2) Improving on the previous example, one can show SAT $\notin \mathsf{DTS}[n^{1.6004}]$. If $\mathsf{NTIME}[n] \subseteq \mathsf{DTS}[n^c]$ and $\sqrt{2} \leq c < 2$, then by applying the Speedup and Slowdown Lemmas appropriately, one can derive:

$$\begin{aligned}\mathsf{DTS}[n^{c^2/2+2}] &\subseteq (\exists\, n^{c^2/2})(\forall\, \log n)\mathsf{DTS}[n^2]\\ &\subseteq (\exists\, n^{c^2/2})(\forall\, \log n)(\forall\, n)(\exists\, \log n)\mathsf{DTS}[n]\\ &= (\exists\, n^{c^2/2})(\forall\, n)(\exists\, \log n)\mathsf{DTS}[n]\\ &\subseteq (\exists\, n^{c^2/2})(\forall\, n)\mathsf{DTS}[n^c]\\ &\subseteq (\exists\, n^{c^2/2})\mathsf{DTS}[n^{c^2}]\\ &\subseteq (\exists\, n^{c^2/2})(\exists\, n^{c^2/2})(\forall\, \log n)\mathsf{DTS}[n^{c^2/2}]\\ &= (\exists\, n^{c^2/2})(\forall\, \log n)\mathsf{DTS}[n^{c^2/2}]\\ &\subseteq (\exists\, n^{c^2/2})\mathsf{DTS}[n^{c^3/2}]\\ &\subseteq \mathsf{DTS}[n^{c^4/2}]\end{aligned}$$

When $c^2/2 + 2 > c^4/2$ (which happens if $c < 1.6004$), we have $\mathsf{DTS}[n^a] \subseteq \mathsf{DTS}[n^{a'}]$ for some $a > a'$. Notice that we do not know if $\mathsf{DTS}[n^a] \nsubseteq \mathsf{DTS}[n^{a'}]$ when $a > a'$, as the space bounds on both sides of the inequality are the same. However one can still show by a translation argument (along the lines of Lemma 4.3, proved in the next section) that either $\mathsf{DTS}[n^a] \nsubseteq \mathsf{DTS}[n^{a'}]$ or $\mathsf{NTIME}[n] \nsubseteq \mathsf{DTS}[n^c]$, concluding the proof.

The above proof was found by a computer program. By "found", we mean that the program applied the Speedup and Slowdown Lemmas in precisely the same order and discovered the same parameter settings, having only minimum knowledge of these Lemmas along with a way to check the validity of the parameters. Moreover, the program verified that the above is the *best possible* alternation-trading proof that applies the Speedup and Slowdown Lemmas for at most 7 times. In the next section, we give a formal definition of "alternation-trading proof" and show what can be done with it.

[4]Suppose $\mathsf{NTIME}[n] \subseteq \mathsf{DTS}[n^c]$ and $\Sigma_2\mathsf{TIME}[n] \subseteq \Pi_2\mathsf{TIME}[n^{1+o(1)}]$. The first assumption, along with the Speedup and Slowdown Lemmas, implies that for every k there's a K satisfying $\Sigma_2\mathsf{TIME}[n^k] \subseteq \mathsf{NTIME}[n^{kc}] \subseteq \Sigma_K\mathsf{TIME}[n]$. But the second assumption implies that $\Sigma_K\mathsf{TIME}[n] = \Sigma_2\mathsf{TIME}[n^{1+o(1)}]$. Hence $\Sigma_2\mathsf{TIME}[n^k] \subseteq \Sigma_2\mathsf{TIME}[n^{1+o(1)}]$, which contradicts the time hierarchy for $\Sigma_2\mathsf{TIME}$. Note that a more generic version of this argument is given in Lemma 4.3.

4. TIME-SPACE LOWER BOUNDS FOR SAT

We start by describing our framework for alternation-trading proofs in the case of time-space lower bounds for $\mathsf{NTIME}[n]$ problems, such as SAT. We shall describe the approach in some detail here; the other lower bound settings discussed in the paper will assume knowledge of this section.

Alternation-trading proofs apply a sequence of "speedup" and "slowdown" lemmas in some order, with the goal of reaching a contradiction by a time hierarchy theorem. We formalize alternation-trading proofs for DTS classes as follows:

Definition 4.1. Let $c > 1$. An *alternation-trading proof for c* is a list of complexity classes of the form:

$$(Q_1\ n^{a_1})^{b_2}(Q_2\ n^{a_2})\cdots^{b_k}(Q_k\ n^{a_k})^{b_{k+1}}\mathsf{DTS}[n^{a_{k+1}}], \tag{1}$$

where $k \geq 0$, $Q_i \in \{\exists,\forall\}$, $Q_i \neq Q_{i+1}$, $a_i \geq 0$, and $b_i \geq 1$, for all i. (When $k = 0$, the class is deterministic.) The items of the list are called *lines of the proof*. Each line is obtained from the previous line by applying either a *speedup rule* or a *slowdown rule*. More precisely, if the ith line is

$$(Q_1\ n^{a_1})^{b_2}(Q_2\ n^{a_2})\cdots^{b_k}(Q_k\ n^{a_k})^{b_{k+1}}\mathsf{DTS}[n^{a_{k+1}}],$$

then the $(i+1)$st line has one of four possible forms:

— *(Speedup Rule 0)* $(Q_k\ n^x)^{\max\{x,1\}}(Q_{k+1}\ n^0)^1\mathsf{DTS}[n^{a_{k+1}-x}]$, for $k = 0$ and $0 \leq x \leq a_{k+1}-1$.
— *(Speedup Rule 1)* $(Q_1\ n^{a_1})^{b_2}(Q_2\ n^{a_2})\cdots^{b_k}(Q_k\ n^{\max\{a_k,x\}})^{\max\{x,b_{k+1}\}}(Q_{k+1}\ n^0)^{b_{k+1}}\mathsf{DTS}[n^{a_{k+1}-x}]$, for $k > 0$ and $0 \leq x \leq a_{k+1}-1$.
— *(Speedup Rule 2)* $(Q_1\ n^{a_1})^{b_2}\cdots^{b_k}(Q_k\ n^{a_k})^{b_{k+1}}(Q_{k+1}\ n^x)^{\max\{x,b_{k+1}\}}(Q_{k+2}\ n^0)^{b_{k+1}}\mathsf{DTS}[n^{a_{k+1}-x}]$, for $k > 0$ and $0 \leq x \leq a_{k+1}-1$.
— *(Slowdown Rule)* $(Q_1\ n^{a_1})^{b_2}(Q_2\ n^{a_2})\cdots^{b_{k-1}}(Q_{k-1}\ n^{a_{k-1}})^{b_k}\mathsf{DTS}[n^{c\cdot\max\{a_{k+1},a_k,b_k,b_{k+1}\}}]$, for $k > 0$.[5]

An alternation-trading proof *shows the implication* $(\mathsf{NTIME}[n] \subseteq \mathsf{DTS}[n^c] \implies A_1 \subseteq A_2)$ if its first line is A_1 and its last line is A_2.

The definition comes directly from the statements of the Speedup Lemma (Lemma 3.3) and Slowdown Lemma (Lemma 3.4) for space-bounded computations. (Note the n^0 in the Speedup Lemma corresponds to $\log n \leq n^{o(1)}$, which is negligible. Polylogarithmic and $n^{o(1)}$ factors do not affect any of the elements in these proofs: the speedup, slowdown, time hierarchies, and reduction to SAT are unaffected by $n^{o(1)}$ factors.) Speedup Rules 0, 1, and 2 are easily verified to be syntactic formulations of the Speedup Lemma, where the DTS part of the sped-up computation only reads two guessed configurations. For instance, Speedup Rule 1 holds, since

$$\begin{aligned}&(Q_1\ n^{a_1})^{b_2}(Q_2\ n^{a_2})\cdots^{b_k}(Q_k\ n^{a_k})^{b_{k+1}}\mathsf{DTS}[n^{a_{k+1}}]\\ \subseteq\ &(Q_1\ n^{a_1})^{b_2}(Q_2\ n^{a_2})\cdots^{b_k}(Q_k\ n^{a_k})^{b_{k+1}}(Q_k\ n^x)^{\max\{b_{k+1},x\}}(Q_{k+1}\ n^0)^{b_{k+1}}\mathsf{DTS}[n^{a_{k+1}}]\\ \subseteq\ &(Q_1\ n^{a_1})^{b_2}(Q_2\ n^{a_2})\cdots^{b_k}(Q_k\ n^{\max\{a_k,x\}})^{\max\{b_{k+1},x\}}(Q_{k+1}\ n^0)^{b_{k+1}}\mathsf{DTS}[n^{a_{k+1}}].\end{aligned}$$

Rule 2 is akin to Rule 1, except that it uses opposite quantifiers in its invocation of the Speedup Lemma. The Slowdown Rule works analogously to Lemma 3.4. It follows that alternation-trading proofs are sound.

Note Speedup Rules 0 and 2 add two quantifier blocks, Speedup Rule 1 adds only one quantifier, and all three rules introduce a parameter x.

4.1. Normal Form for Alternation-Trading Proofs

We now introduce a *normal form* for alternation-trading proofs. Define any class of the form (1) to be *simple*. We show that any lower bound provable with complementary simple classes can also be

[5]Strictly speaking, it is redundant to put b_k in the maximum here, as b_k is always dominated by the other values. We put it in the rule for the sake of completeness.

established with a normal form proof. This greatly reduces the degrees of freedom in a proof, as we no longer have to worry about *which* complementary simple classes to choose for the contradiction.

Define classes A_1 and A_2 to be *complementary* if A_1 is the class of complements of languages in A_2. Every known (model-independent) time-space lower bound for SAT shows "$\mathsf{NTIME}[n] \subseteq \mathsf{DTS}[n^c]$ implies $A_1 \subseteq A_2$", for some complementary simple classes A_1 and A_2, contradicting a time hierarchy theorem (in particular, Theorem 3.5). A similar claim holds for nondeterministic time-space lower bounds against tautologies (which prove $\mathsf{NTIME}[n] \subseteq \mathsf{coNTS}[n^c]$ implies $A_1 \subseteq A_2$), for d-dimensional machine lower bounds solving SAT (which prove $\mathsf{NTIME}[n] \subseteq \mathsf{DTIME}_d[n^c]$ implies $A_1 \subseteq A_2$), and other related problems such as those described in the Introduction.

Definition 4.2. Let $c \geq 1$. An alternation-trading proof for c is in *normal form* if (a) the first and last lines are $\mathsf{DTS}[n^a]$ and $\mathsf{DTS}[n^{a'}]$ respectively, for some $a \geq a'$, and (b) no other lines are DTS classes.

Next we show that a normal form proof for c implies that $\mathsf{NTIME}[n] \nsubseteq \mathsf{DTS}[n^c]$, and that any alternation-trading proof of $\mathsf{NTIME}[n] \nsubseteq \mathsf{DTS}[n^c]$ can be rewritten in normal form.

LEMMA 4.3. *Let $c \geq 1$. If there is an alternation-trading proof for c in normal form having at least two lines, then* $\mathsf{NTIME}[n] \nsubseteq \mathsf{DTS}[n^c]$.

PROOF. Let P be an alternation-trading proof for c in normal form. We consider two cases.

Suppose $a > a'$. In this case, $\mathsf{NTIME}[n] \subseteq \mathsf{DTS}[n^c]$ implies $\mathsf{DTS}[n^a] \subseteq \mathsf{DTS}[n^{a-\delta}]$ for some $\delta > 0$. By translation, $\mathsf{DTS}[n^a] \subseteq \mathsf{DTS}[n^{a-\delta}]$ implies

$$\mathsf{DTS}[n^{a^2/(a-\delta)}] \subseteq \mathsf{DTS}[n^a] \subseteq \mathsf{DTS}[n^{a-\delta}],$$

and $\mathsf{DTS}[n^{a\cdot(a/(a-\delta))^i}] \subseteq \mathsf{DTS}[n^{a-\delta}]$ for all $i \geq 0$. Since $\delta > 0$, this implies $\mathsf{DTS}[n^L] \subseteq \mathsf{DTS}[n^{a-\delta}]$ for all $L \geq a-\delta$. Therefore, if $\mathsf{NTIME}[n] \subseteq \mathsf{DTS}[n^c]$ then for all $L \geq a$,

$$\mathsf{NTIME}[n^L] \subseteq \mathsf{DTS}[n^{Lc}] \subseteq \mathsf{DTS}[n^{a-\delta}] \subseteq \mathsf{coNTIME}[n^{a-\delta}],$$

a contradiction to the time hierarchy (Theorem 3.5).

Suppose $a = a'$. Let A be a line in P with a positive number of alternations. (Such a line must exist since P has at least two lines.) The proof P proves that $\mathsf{NTIME}[n] \subseteq \mathsf{DTS}[n^c]$ implies $\mathsf{DTS}[n^a] \subseteq A \subseteq \mathsf{DTS}[n^{a'}]$, so $A = \mathsf{DTS}[n^a]$.

Since $\mathsf{DTS}[n^a]$ is closed under complement and $A = \mathsf{DTS}[n^a]$, we have

$$A = A', \tag{2}$$

where A' is the complement of A. Without loss of generality, assume the first quantifier block of A is existential. Then for some sufficiently small $\delta \in (0,1]$, we have

$$A' = (\forall\, n^\delta)A' \text{ and } A = (\exists\, n^\delta)A. \tag{3}$$

Now consider the class $\mathsf{DTS}[n^{\delta\lceil\frac{k}{\delta}\rceil}] \supseteq \mathsf{DTS}[n^k]$, for arbitrary $k \geq 1$. By the Speedup Lemma (Lemma 3.3) and the fact that $\mathsf{DTS}[n^a] \subseteq A'$ for some $a > 0$,

$$\mathsf{DTS}[n^k] \subseteq \mathsf{DTS}[n^{\delta\lceil\frac{k}{\delta}\rceil}] \subseteq \underbrace{(\exists\, n^\delta)(\forall\, n^\delta)\cdots(\exists\, n^\delta)(\forall\, n^\delta)}_{\lceil k/\delta\rceil}A'.$$

By repeatedly applying equations (2) and (3) and using the fact that $\delta < 1$, we derive

$$\begin{aligned}(\exists\, n^\delta)(\forall\, n^\delta)\cdots(\exists\, n^\delta)(\forall\, n^\delta)A' &= (\exists\, n^\delta)(\forall\, n^\delta)\cdots(\exists\, n^\delta)A' \\ &= (\exists\, n^\delta)(\forall\, n^\delta)\cdots(\exists\, n^\delta)A \\ &= (\exists\, n^\delta)(\forall\, n^\delta)\cdots A \\ &= \cdots = (\exists\, n^\delta)(\forall\, n^\delta)A' = (\exists\, n^\delta)A' = (\exists\, n^\delta)A = A.\end{aligned}$$

Therefore $\mathsf{DTS}[n^k] \subseteq A$, for *every* $k \geq 1$. Assuming $\mathsf{NTIME}[n] \subseteq \mathsf{DTS}[n^c]$, we have $\mathsf{NP} \subseteq \bigcup_{k \geq 1} \mathsf{DTS}[n^k]$. Hence

$$\mathsf{NP} \subseteq \bigcup_{k \geq 1} \mathsf{DTS}[n^k] \subseteq A = \mathsf{DTS}[n^a] \subseteq \mathsf{coNTIME}[n^a],$$

contradicting the time hierarchy (Theorem 3.5). Therefore $\mathsf{NTIME}[n] \not\subseteq \mathsf{DTS}[n^c]$ in this case as well. □

THEOREM 4.4. *Let A_1 and A_2 be complementary. If there is an alternation-trading proof P for c that shows* ($\mathsf{NTIME}[n] \subseteq \mathsf{DTS}[n^c] \Longrightarrow A_1 \subseteq A_2$)*, then there is a normal form proof for c, of length at most that of P.*

PROOF. Consider an alternation-trading proof P for c, written as

$$P = A_1, C_1, \ldots, C_k, A_2.$$

Define the *dual proof P'* by

$$P' = A_2, \neg C_1, \ldots, \neg C_k, A_1,$$

where the notation $\neg C$ denotes the unique complementary simple class for C, *i.e.* every '$\forall$' in C is replaced with '$\exists$', and vice-versa. Note that P' is an alternation-trading proof if and only if P is one.

Since the quantifiers of the first and last line of P are different, note that there must be a line $C_i = \mathsf{DTS}[n^a]$ for some a.

— Suppose there is only one deterministic class in P; call it C_i. Then

$$P'' = C_i, C_{i+1}, \ldots C_k, A_2, \neg C_1, \ldots, \neg C_i$$

is also an alternation-trading proof, obtained by piecing together the appropriate lines from P and P'. However, $C_i = \neg C_i$, since $\mathsf{DTS}[n^a]$ is closed under complement. Hence P'' is in normal form: its first and last lines are DTS classes, and no intermediate class is a DTS class.

— Suppose there are $k \geq 2$ different DTS classes in P. Write P as

$$P = A_1, \ldots, \mathsf{DTS}[n^{a_1}], \ldots, \mathsf{DTS}[n^{a_2}], \ldots, \ldots, \mathsf{DTS}[n^{a_k}], \ldots, A_2.$$

There are two cases:

- If there is an $i \in [k]$ satisfying $a_i \geq a_{i+1}$, we are done: let P'' to be the sequence of lines from $\mathsf{DTS}[n^{a_i}]$ and $\mathsf{DTS}[n^{a_{i+1}}]$, and this is in normal form.
- If $a_i < a_{i+1}$ for every i, then set $P'' = \mathsf{DTS}[n^{a_k}], \ldots, A_2, \ldots, \mathsf{DTS}[n^{a_1}]$, where the classes in the first "…" in P'' are taken directly from P, and the classes in the second "…" in P'' are obtained by taking the lines $A_2, \ldots, \mathsf{DTS}[n^{a_1}]$ in P'. P'' is in normal form since $a_k > a_1$.

□

For the remainder of this section, we assume that all alternation-trading proofs under discussion are in normal form. The important consequence of Lemma 4.3 and Theorem 4.4 is that we greatly simplify the space of lower bound proofs we have to consider. For example, using normal form proofs, we can show that every application of Speedup Rule 2 can always be replaced by applications of Speedup Rule 1. For this reason we just refer to *the Speedup Rule*, depending on which of Rule 0 or Rule 1 applies. This further simplifies the kinds of proofs we have to study.

LEMMA 4.5. *For every alternation-trading proof that* $\mathsf{NTIME}[n] \subseteq \mathsf{DTS}[n^c] \Longrightarrow A_1 \subseteq A_2$*, there is another alternation-trading proof of the same implication that does not use Speedup Rule 2.*

PROOF. Consider a proof P that applies Speedup Rule 2 at some line. The previous line has the form

$$B = (Q_1\, n^{a_1})^{b_2} \cdots^{b_k} (Q_k\, n^{a_k})^{b_{k+1}} \mathsf{DTS}[n^{a_{k+1}}]$$

and the next line has the form

$$A = (Q_1\ n^{a_1})^{b_2} \cdots^{b_k} (Q_k\ n^{a_k})^{b_{k+1}} (Q_{k+1}\ n^{x'})^{\max\{x', b_{k+1}\}} (Q_{k+2}\ n^0)^{b_{k+1}} \mathsf{DTS}[n^{a_{k+1}-x'}],$$

for some $x' \in [0, a_{k+1}-1]$. Starting at the line B, let us apply Speedup Rule 1 instead, with $x = 0$. This yields the class

$$B_1 = (Q_1\ n^{a_1})^{b_2} \cdots^{b_k} (Q_k\ n^{\max\{a_k,0\}})^{\max\{b_{k+1},0\}} (Q_{k+1}\ n^0)^{b_{k+1}} \mathsf{DTS}[n^{a_{k+1}-0}]$$

$$= (Q_1\ n^{a_1})^{b_2} \cdots^{b_k} (Q_k\ n^{a_k})^{b_{k+1}} (Q_{k+1}\ n^0)^{b_{k+1}} \mathsf{DTS}[n^{a_{k+1}}].$$

Applying Speedup Rule 1 with $x = x'$, we obtain the class

$$(Q_1\ n^{a_1})^{b_2} (Q_2\ n^{a_2}) \cdots^{b_k} (Q_k\ n^{a_k})^{b_{k+1}} (Q_{k+1}\ n^{\max\{x',0\}})^{\max\{x', b_{k+1}\}} (Q_{k+2}\ n^0)^{b_{k+1}} \mathsf{DTS}[n^{a_{k+1}-x'}],$$

which is precisely the class A obtained above. □

Proof Annotations. Different lower bound proofs can result in quite different sequences of speedups and slowdown rule applications. A *proof annotation* represents such a sequence.

Definition 4.6. A *proof annotation* for an alternation-trading proof of ℓ lines is the $(\ell-1)$-bit vector A where for all $i = 1, \ldots, \ell-1$, $A[i] = 1$ (respectively, $A[i] = 0$) if the ith line applies a Speedup Rule (respectively, a Slowdown Rule).

An $(\ell-1)$-bit proof annotation corresponds to a "strategy" for an ℓ-line proof. For a normal form alternation-trading proof with ℓ lines, it is not hard to show that its annotation A must have $A[1] = 1$, $A[\ell-2] = 0$, and $A[\ell-1] = 0$.

The number of possible normal form proof annotations is closely related to the number of well-balanced strings over parentheses. Recall that the kth Catalan number is $C(k) = \frac{1}{k+1}\binom{2k}{k}$. A well-known fact states that the number of well-balanced strings of length $2k$ is $C(k)$.

PROPOSITION 4.7. *Let $\ell > 3$ be even. The number of possible annotations for normal form proofs of ℓ lines is $C(\ell/2-1)$.*

PROOF. For clarity, let us denote the left-parenthesis character by L, and the right-parenthesis by R. We claim that the set of normal form annotations with ℓ lines can be put in one-to-one correspondence with strings of balanced parentheses of the form LxR, where x is an non-empty balanced parentheses string of length $\ell-2$. Therefore, there is a one-to-one correspondence between normal form annotations of ℓ lines and well-parenthesized strings of length $\ell-2$, which will establish the proposition.

Each normal form annotation A of ℓ lines begins with $A[1] = 1$, ends with $A[\ell-2] = A[\ell-1] = 0$, and has the property that for all $i = 2, \ldots, \ell-1$,

$$\sum_{j=1}^{i} A[j] \geq \left(\sum_{j=1}^{i} (1 - A[j]) \right) - 1. \tag{4}$$

That is, the sum of all ones among $A[1], \ldots, A[i]$ is at least the sum of all zeroes among $A[1], \ldots, A[i]$, minus one. Furthermore, the sum of all ones in A equals the sum of all zeroes in A minus one. (This follows from the facts: a normal form proof must have a nonzero number of quantifiers on each line, for all lines other than the first and the last; two quantifiers are introduced with the first speedup; one quantifier is introduced in every subsequent speedup; one quantifier is removed with each slowdown.)

We can map A to a string of parentheses B, by defining a map $\mu(0) = R$, $\mu(1) = L$, and setting

$$B = L\ \mu(A[1])\mu(A[2]) \cdots \mu(A[\ell-2])R.$$

Note that B is a string of length ℓ. We now claim that B is well-balanced. Let $B[i]$ denote the ith character in B. By the inequality (4) and the fact that $A[\ell-1] = 0$, we have that for all $i = 1, \ldots, \ell$, the sum of all L's among $B[1],\ldots,B[i]$ is at least the sum of all R's among $B[1],\ldots,B[i]$, and the total number of L's among $B[1],\ldots,B[\ell]$ equals the total number of R's among $B[1],\ldots,B[\ell]$. This is exactly the (iterative) definition of a well-balanced parenthesis string.

Therefore, normal form annotations with ℓ lines are in one-to-one correspondence with strings of balanced parentheses of length ℓ of the form LxR. The number of these strings is $C(\ell/2-1)$. □

Proposition 4.7 implies that the number of possible annotations for proofs of ℓ lines is $\Theta(2^{\ell}/\ell^{3/2})$. Note that an annotation *does not* determine a proof entirely, as there are other parameters to set. (The problem of determining optimal values for these parameters is tackled in the next section.) To illustrate the annotation concept, we give four examples.

— Lipton-Viglas' $n^{\sqrt{2}}$ lower bound [Lipton and Viglas 1999] (from the Introduction) has the annotation $[1,0,0]$.
— The $n^{1.6004}$ bound of the *Short Introduction* (Section 3) corresponds to $[1,1,0,0,1,0,0]$.
— The n^{ϕ} bound of Fortnow-Van Melkebeek [Fortnow and van Melkebeek 2000] is an inductive proof, corresponding to an infinite sequence of annotations. In normal form, the sequence is: $[1,0,0],[1,1,0,0,0],[1,1,1,0,0,0,0],\ldots$
— The $n^{2\cos(\pi/7)}$ bound [Williams 2008] has two stages of induction. Let $A=1,0,\ldots,1,0,0$, where the '$\ldots$' contain any number of repetitions of the pattern $1,0$. The sequence of annotations is $[A]$, $[1,A,A]$, $[1,1,A,A,A]$, $[1,1,1,A,A,A,A],\ldots$
That is, the proof performs many speedups, then a sequence of many slowdown-speedup alternations, then two consecutive slowdowns, repeating this until all the quantifiers have been removed.

4.2. Translation To Linear Programming

Given a (normal form) proof annotation, how can we determine the best proof possible with it? That is, how do we determine the largest time lower bound that can be obtained by following the annotation? For a fixed annotation, the runtimes of the first and last DTS classes of the proof are free parameters, and each application of a Speedup Rule also introduces a parameter x_i. We now show how to reduce an annotation A and $c>1$ to a linear program that is feasible if and only if there is an alternation-trading proof of $\mathsf{NTIME}[n]\not\subseteq\mathsf{DTS}[n^c]$ with annotation A. More precisely, the problem of setting parameters can be viewed as an arithmetic circuit evaluation where the circuit has max gates, addition gates, and input gates that multiply their input by c. Such circuits can be evaluated using a linear program (cf. [Derman 1972]) that minimizes the sum of the gate values.

Let A be an annotation of $\ell-1$ bits, and let m be the maximum number of quantifier blocks in a line of A; note m is easily computed in linear time. The target LP has variables $a_{i,j}$, $b_{i,j}$, and x_i, for all $i=0,\ldots,\ell-1$ and $j=1,\ldots,m$. The variables $a_{i,j}$ represent the runtime exponent of the jth quantifier block in the class on the ith line (where the innermost quantifier block is considered "first", the next innermost is considered "second", and so on), $b_{i,j}$ is the input exponent to the jth quantifier block of the class on the ith line, and for all lines i that use a Speedup Rule, x_i is the choice of x in the Speedup Rule. For example:

— If the kth line of a proof is $\mathsf{DTS}[n^a]$, the corresponding constraints are
$$a_{k,1}=a,\ \ b_{k,1}=1,\ \ (\forall k>0)\ a_{k,i}=b_{k,i}=0.$$
— If the kth line of a proof is $(\forall\, n^{a''})^{b'}(\exists\, n^{a'})^{b}\mathsf{DTS}[n^a]$, then the constraints are
$$a_{k,0}=a,\ \ b_{k,1}=b,\ \ a_{k,1}=a',\ \ b_{k,1}=b',\ \ a_{k,2}=a'',\ \ b_{k,2}=1,\ \ (\forall k>2)\ a_{k,i}=b_{k,i}=0.$$

The objective is to minimize $\sum_{i,j}(a_{i,j}+b_{i,j})+\sum_i x_i$. The LP constraints depend on the lines of the annotation, as follows.

Initial Constraints. For the 0th and $(\ell-1)$th lines we have $a_{0,1}\geq a_{\ell-1,1}$, as well as

$$a_{0,1}\geq 1,\ b_{0,1}=1,\ \ (\forall\, k>1)\ a_{0,k}=b_{0,k}=0,\ \text{ and }\ a_{\ell-1,1}\geq 1,\ b_{\ell-1,1}=1,\ \ (\forall k>1)\ a_{\ell-1,k}=b_{\ell-1,k}=0,$$

representing $\mathsf{DTS}[n^{a_{0,1}}]$ and $\mathsf{DTS}[n^{a_{\ell-1,0}}]$, respectively.

The first line of a proof always applies Speedup Rule 1, having the form $(Q_1 n^x)^{\max\{x,1\}}(Q_2\, n^0)^1 \mathsf{DTS}[n^{a-x}]$. So the constraints for the 1st line are:

$$a_{1,1} = a_{0,1} - x_1,\ b_{1,1} = 1,\ a_{1,2} = 0,\ b_{1,2} \geq x_1,\ b_{1,2} \geq 1,\ a_{1,3} = x_3,\ b_{1,3} = 1,$$
$$(\forall\, k:\ 4 \leq k \leq m)\ a_{1,k} = b_{1,k} = 0.$$

The below constraint sets simulate the Speedup and Slowdown Rules:

Speedup Rule Constraints. For the ith line where $i > 1$ and $A[i] = 1$, the constraints are

$$a_{i,1} \geq 1,\ a_{i,1} \geq a_{i-1,1} - x_i,\ b_{i,1} = b_{i-1,1},\ a_{i,2} = 0,\ b_{i,2} \geq x_i,\ b_{i,2} \geq b_{i-1,1},\ a_{i,3} \geq a_{i-1,2},$$
$$a_{i,3} \geq x_i,\ b_{i,3} = b_{i-1,2},\ (\forall\, k:\ 4 \leq k \leq m)\ a_{i,k} = a_{i-1,k-1}, b_{i,k} = b_{i-1,k-1}.$$

These constraints express that $\cdots\ ^{b_2}(Q_2\, n^{a_2})^{b_1} \mathsf{DTS}[n^{a_1}]$ in the $(i-1)$th line is replaced with

$$\cdots\ ^{b_2}(Q_2\, n^{\max\{a_2,x\}})^{\max\{x,b_1\}}(Q_1\, n^0)^{b_1} \mathsf{DTS}[n^{\max\{a_1-x,1\}}]$$

in the ith line, where Q_1 is opposite to Q_2.

Slowdown Rule Constraints. For the ith line where $A[i] = 0$, the constraints are

$$a_{i,1} \geq c \cdot a_{i-1,1},\ a_{i,1} \geq c \cdot a_{i-1,2},\ a_{i,1} \geq c \cdot b_{i-1,1},\ a_{i,1} \geq c \cdot b_{i-1,2},\ b_{i,1} = b_{i-1,2}$$
$$(\forall\, k:\ 2 \leq k \leq m-1)\ a_{i,k} = a_{i-1,k+1},\ b_{i,k} = b_{i-1,k+1},\ a_{i,m} = b_{i,m} = 0.$$

These express the replacement of $\cdots\ ^{b_2}(Q_1 n^{a_2})^{b_1} \mathsf{DTS}[n^{a_1}]$ in the $(i-1)$th line with

$$\cdots\ ^{b_2} \mathsf{DTS}[n^{c \cdot \max\{a_1,a_2,b_1,b_2\}}]$$

in the ith line.

This concludes the description of the LP. To find the largest c that still yields a feasible LP, we can simply binary search for it. The following theorem summarizes the above discussion, along with the fact that linear programming on n variables, $\text{poly}(n)$ constraints, and n-bit coefficients can be solved in $\text{poly}(n)$ time.

THEOREM 4.8. *Given a proof annotation of n lines, the best possible lower bound proof following the annotation can be determined up to n digits of precision, in* $\text{poly}(n)$ *time.*

4.3. Empirical Results

Following the above formulation, we wrote a computer program that given a proof annotation generates the relevant linear programming instance, solves it, then prints the resulting SAT time-space lower bound proof in human-readable form. For proof annotations exceeding 100 lines, we used `lp_solve` to solve the corresponding linear program.[6]

To search for proof annotations, we wrote routines for exhaustive search and uniform random generation of annotations. To perform the sampling, we adapted a simple method for producing random well-balanced strings, given by Arnold and Sleep [Arnold and Sleep 1980]. We also wrote heuristic search programs that derive new proofs from old ones. One program starts with a queue of annotations, pulls the head of the queue, and tries all ways to add at most four bits to the annotation. If the resulting lower bound from the new annotation increases, the new annotation is added to the tail of the queue. This simple strategy generated all the optimal lower bounds that were found by exhaustive search, and more.[7] (By "optimal", we mean "optimal with respect to the length of the proof.")

Using up to 20 digits of precision, we verified all the previously known lower bounds, such as the $n^{2\cos(\pi/7)}$ bound. In some cases, we found shorter-length lower bound proofs than previously

[6]The `lp_solve` package is an open source simplex-based linear programming solver, available to download at `http://lpsolve.sourceforge.net/`.
[7]Our Maple code can be found at `http://www.stanford.edu/rrwill/LB.*` where `*` can be `txt` (text file) or `mws` (Maple worksheet).

known, but no lower bound greater than $n^{2\cos(\pi/7)}$. To check if the choice of parameters in the $2\cos(\pi/7) \approx 1.8019$ lower bound proof were tight, we tested a LP corresponding to a 300-line annotation for our proof of the 1.8019 bound, and found that it was feasible for $c = 1.8017$ but infeasible for 1.8018; moreover, its choice of parameters corresponded precisely to those of previous work [Williams 2008]. For all even $k = 2,\ldots,26$, we exhaustively searched over all valid proof annotations with k lines. The best lower bounds found for each length k are given in the below table. For $k > 26$ we have not exhaustively searched all proofs, but instead used a heuristic search as described above; these rows of the table are marked with an asterisk.[8]

k	Optimal Proof Annotation(s) With k Lines	L.B. Exponent
4	[1,0,0]	1.4142
6	[1,0,1,0,0]	1.5213
	[1,1,0,0,0]	
8	[1,1,0,0,1,0,0]	1.6004
10	[1,1,0,0,1,0,1,0,0]	1.6333
	[1,1,0,1,0,0,1,0,0]	
	[1,1,1,0,0,0,1,0,0]	
12	[1,1,1,0,0,1,0,0,1,0,0]	1.6635
14	[1,1,1,0,0,1,0,0,1,0,1,0,0]	1.6871
16	[1,1,1,0,0,1,0,1,0,0,1,0,1,0,0]	1.6996
	[1,1,1,0,1,0,0,1,0,0,1,0,1,0,0]	
	[1,1,1,1,0,0,0,1,0,0,1,0,1,0,0]	
18	[1,1,1,1,0,0,1,0,0,1,0,0,1,0,1,0,0]	1.7121
20	[1,1,1,1,0,0,1,0,0,1,0,1,0,0,1,0,1,0,0]	1.7232
22	[1,1,1,1,0,0,1,0,0,1,0,1,0,0,1,0,1,0,1,0,0]	1.7322
24	[1,1,1,1,0,0,1,0,1,0,0,1,0,1,0,0,1,0,1,0,1,0,0]	1.7378
	[1,1,1,1,0,1,0,0,1,0,0,1,0,1,0,0,1,0,1,0,1,0,0]	
	[1,1,1,1,1,0,0,0,1,0,0,1,0,1,0,0,1,0,1,0,1,0,0]	
26	[1,1,1,1,1,0,0,1,0,0,1,0,0,1,0,1,0,0,1,0,1,0,1,0,0]	1.7437
28*	[1,1,1,1,1,0,0,1,0,0,1,0,1,0,0,1,0,1,0,0,1,0,1,0,1,0,0]	1.7491
30*	[1,1,1,1,1,0,0,1,0,0,1,0,1,0,0,1,0,1,0,1,0,0,1,0,1,0,1,0,0]	1.7537
32*	[1,1,1,1,1,0,0,1,0,0,1,0,1,0,0,1,0,1,0,1,0,0,1,0,1,0,1,0,1,0,0]	1.7577
34*	[1,1,1,1,1,0,0,1,0,1,0,0,1,0,1,0,0,1,0,1,0,1,0,0,1,0,1,0,1,0,1,0,0]	1.7606
	[1,1,1,1,1,0,1,0,0,1,0,0,1,0,1,0,0,1,0,1,0,1,0,0,1,0,1,0,1,0,1,0,0]	
	[1,1,1,1,1,1,0,0,0,1,0,0,1,0,1,0,0,1,0,1,0,1,0,0,1,0,1,0,1,0,1,0,0]	

The proofs produced by the annotations in the table have strong similarities to those in the $2\cos(\pi/7)$ lower bound. For example, the best 14-line proof (proving an $\Omega(n^{1.6871})$ lower bound) looks like:

```
0, DTS[n^5.275587925]
1, (E n^1.853485593)(A n^1.)DTS[n^3.422102331]
2, (E n^1.853485593)(A n^1.422102331)(E n^1.)DTS[n^2.]
3, (E n^1.853485593)(A n^1.422102331)(E n^1.)(A n^1.)DTS[n^1.]
4, (E n^1.853485593)(A n^1.422102331)(E n^1.)DTS[n^1.687100000]
5, (E n^1.853485593)(A n^1.422102331)DTS[n^2.846306408]
6, (E n^1.853485593)(A n^1.423153204)(E n^1.)DTS[n^1.423153204]
7, (E n^1.853485593)(A n^1.423153204)DTS[n^2.401001771]
8, (E n^1.853485593)DTS[n^4.050730087]
9, (E n^1.853485593)(A n^1.)DTS[n^2.197244494]
10, (E n^1.853485593)DTS[n^3.706971186]
11, (E n^1.853485593)(A n^1.)DTS[n^1.853485593]
12, (E n^1.853485593)DTS[n^3.127015544]
13, DTS[n^5.275587925]
```

[8]Note: we did not formally verify that (for example) the three normal form annotations with 10 lines actually yield the exact same lower bound exponent, nor did we verify this for any of the other annotations that yield approximately the same exponent.

It is clear that there is a strong correlation between later rows of the table and earlier ones. For example, there is a tie for best annotation at 10, 16, 24, and 34 lines, among three annotations that differ only in three bits. To further understand what is happening, let us introduce some abbreviations. Where an annotation contains the string $(1\ 0)^k\ 0$, we put the symbol **k**, for $k \geq 1$. Where an annotation contains the string 11000, we just put **0**. The following table emerges:

#Lines	Best Proof Annotation(s)	L.B.	Δ
4	**1**	1.4142	0
6	**2**	1.5213	0.1071
	0		
8	1 **2**	1.6004	0.0791
10	1 **1 2**	1.633315	0.032915
	1 **2 1**		
	1 **0 1**		
12	1 1 **1 1 1**	1.6635	0.0302
14	1 1 **1 1 2**	1.6871	0.0236
16	1 1 **1 2 2**	1.699676	0.012576
	1 1 **2 1 2**		
	1 1 **0 1 2**		
18	1 1 1 **1 1 1 2**	1.7121	0.0125
20	1 1 1 **1 1 2 2**	1.7232	0.0111
22	1 1 1 **1 1 2 3**	1.7322	0.0090
24	1 1 1 **1 2 2 3**	1.737851	0.005651
	1 1 1 **2 1 2 3**		
	1 1 1 **0 1 2 3**		
26	1 1 1 1 **1 1 1 2 3**	1.7437	0.005849
28*	1 1 1 1 **1 1 2 2 3**	1.7491	0.0054
30*	1 1 1 1 **1 1 2 3 3**	1.7537	0.0046
32*	1 1 1 1 **1 1 2 3 4**	1.7577	0.0040
34*	1 1 1 1 **1 2 2 3 4**	1.760632	0.002932
	1 1 1 1 **2 1 2 3 4**		
	1 1 1 1 **0 1 2 3 4**		

(Here, the Δ of a row is the difference between the exponent of that row and the exponent of the previous row.) For an optimal annotation that ends with a non-zero **k**, a longer optimal annotation can be obtained by adding either a **k** or **k+1** to the end, and a 1 at the beginning. (There are of course some restrictions– there are no more than three consecutive **1**'s, no more than two consecutive **2**'s, *etc.*) This table suggests that we examine proof annotations of the form $1\ \cdots\ 1$ **0 1 2 3 4**$\cdots$. Unfortunately these annotations do not lead to an improvement over the $2\cos(\pi/7)$ exponent. To illustrate, for the 424 line proof annotation denoted by

$$1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ \mathbf{0\ 1\ 2\ 3\ 4}\ \cdots\ \mathbf{17\ 18\ 19},$$

experiments with `lp_solve` revealed that the optimal exponent is only in the interval $[1.80175, 1.8018)$. These results point strongly to the idea that there is no alternation-trading proof that $\mathsf{NTIME}[n] \not\subseteq \mathsf{DTS}[n^c]$, for any $c > 2\cos(\pi/7) \approx 1.8019$. We are led to:

CONJECTURE 4.9. *There is no alternation-trading proof that* $\mathsf{NTIME}[n] \not\subseteq \mathsf{DTS}[n^c]$*, for any* $c > 2\cos(\pi/7)$.

This conjecture was recently settled by Sam Buss and the author [Buss and Williams 2012].

4.4. Provable Limit on Alternation-Trading Proofs

The proof of Conjecture 4.9 is rather technical, building substantially on the work of the present paper. In brief, the proof of Conjecture 4.9 performs a series of reductions on the rules of alternation-trading proofs, simplifying the rules further and further (introducing some new rules which capture countably many applications of the original rules), until a proof system is reached that can be completely analyzed. It is much easier to prove a partial result along these lines:

THEOREM 4.10. *There is no alternation-trading proof that* $\mathsf{NTIME}[n] \not\subseteq \mathsf{DTS}[n^2]$.

In their paper showing that SAT cannot be solved in $O(n^{\phi})$ time and $n^{o(1)}$ space, Fortnow *et al.* [Fortnow et al. 2005] write that "some complexity theorists feel that improving the golden ratio exponent beyond 2 would require a breakthrough." The below is a simple formal proof of this sentiment.

Proof of Theorem 4.10. (Sketch) Suppose there is such a proof, and let A be a minimum length annotation in normal form for it. We claim that A can be made shorter, yet the resulting LP is still feasible if the original LP was feasible.

First we observe that every normal form annotation contains a sequence $1,0$. Recall that in the proof of Proposition 4.7, we showed that normal form annotations can be put in 1-1 correspondence with strings of balanced parentheses of the form `(`x`)`, where x is an non-empty balanced parentheses string. The first speedup in a proof corresponds to `((`, as it introduces two quantifiers, all other speedup applications correspond to a `(`, and a slowdown correponds to a `)`. Since there is always an adjacent parentheses pair `()` in any string of balanced parentheses, there must also be some occurrence of $1,0$ in a valid proof annotation. If this $1,0$ can be removed from A without changing the feasibility of the underlying linear program, the claim is proved.

The two lines in the proof corresponding to the sequence $1,0$ (including the previous line) have the form:

$$\cdots^{b_{k-1}} (Q_{k-1}\ n^{a_{k-1}})^{b_k} (Q_k\ n^{a_k})^{b_{k+1}} \mathsf{DTS}[n^{a_{k+1}}] \tag{5}$$

$$\cdots^{b_{k-1}} (Q_{k-1}\ n^{a_{k-1}})^{b_k} (Q_k\ n^{\max\{x,a_k\}})^{\max\{x,b_{k+1}\}} (Q_{k+1}\ \log n^0)^{b_{k+1}} \mathsf{DTS}[n^{a_{k+1}-x}] \tag{6}$$

$$\cdots^{b_{k-1}} (Q_{k-1}\ n^{a_{k-1}})^{b_k} (Q_k\ n^{\max\{x,a_k\}})^{\max\{x,b_{k+1}\}} \mathsf{DTS}[n^{\max\{c(a_{k+1}-x),cx,cb_{k+1}\}}] \tag{7}$$

Every parameter in the class (7) is at least the corresponding parameter in the class (5), except for possibly the runtime of the DTS computation. Hence if $a_{k+1} \le c(a_{k+1}-x)$, or $a_{k+1} \le cx$, or $a_{k+1} \le cb_{k+1}$, then $1,0$ could be removed without changing the feasibility of the LP. However, if both $a_{k+1} > c(a_{k+1}-x)$ and $a_{k+1} > cx$, then $2a_{k+1} > c(a_{k+1}-x)+cx$, which implies a contradiction for $c \ge 2$. □

4.5. Generalization to Arbitrary Sublinear Space Bounds

While studying the output of the theorem prover did not yield an improved time lower bound, it did provide enough insight to prove a new lower boundon the time-space product of any SAT algorithm.

Using Lemma 3.3 in its full generality, it is possible to adapt our linear programming formulation to prove time lower bounds for SAT for any fixed space bound n^{δ}, where $\delta \in (0,1)$. This introduces a new variable δ in the LP formulation. Table I gives time-space pairs for which our theorem prover has shown that no SAT algorithm can satisfy both time and space requirements simultaneously.

Based on this table, it is natural to conjecture that the time-space product for any algorithm solving SAT is at least $\Omega(n^{1.75})$.[9]

In what follows, we prove a weaker time space product lower bound of $\Omega(n^{1.618})$, and show how one can extend this proof to yield stronger results for specific time and space bounds. The previously best known bound on the time-space product was $\Omega(n^{1.573})$ [Fortnow et al. 2005]. While the proof annotations in the below are analogous to the $2\cos(\pi/7)$ bound (as suggested by the experiments),

[9]An earlier version of the paper conjectured (and unfortunately claimed) that the time-space product for *all* time and space bounds was at least $n^{1.801}$. Subsequent work with Sam Buss has since demonstrated that proving that conjecture would require new techniques [Buss and Williams 2012]. This subsequent work also gives a new way of automatically proving time-space lower bounds, which further improve the tradeoffs in the above table.

Table I. SAT Time-Space Lower Bounds

Time	Space
$n^{1.06}$	$n^{.9}$
$n^{1.14}$	$n^{.75}$
$n^{1.24}$	$n^{.666}$
$n^{1.32}$	$n^{.5}$
$n^{1.44}$	$n^{.333}$
$n^{1.51}$	$n^{.25}$
$n^{1.65}$	$n^{.1}$
$n^{1.75}$	$n^{.05}$

the parameter settings in the below proof rely a great deal on our study of the theorem prover's output.

THEOREM 4.11. *Let $t(n)$ and $s(n)$ be bounded above by polynomials. Any algorithm solving SAT in time t and space s requires $t \cdot s = \Omega(n^{\phi-\varepsilon})$ for all $\varepsilon > 0$.*

PROOF. Suppose SAT is solved in time $t = n^c$ and space $s = n^d$, with $c+d \le \phi = 1.618\cdots$, where ϕ is the golden ratio. Of course we must have $c \ge 1$, and so $d \le 1-\phi < 1$. By Theorem 3.1, it follows that $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^{c+o(1)}, n^{d+o(1)}]$.

In this setting, we will introduce a new parameter $e \in [0, 2-\phi]$ representing the space exponent of the initial DTISP computation. To obtain a contradiction, the parameter e will need to exceed the final space exponent obtained in the proof derivation, in order to yield a contradiction. More precisely, we will give a proof of the form

$$\mathsf{DTISP}[n^C, n^e] \subseteq \cdots \subseteq \mathsf{DTISP}[n^D, n^{e'}],$$

where C, D, e' are rational functions of c and d; to obtain a contradiction we will require that both $C > D$ and $e > e'$.

Define the sequences

$$c_1 := 2-e-d,\ c_{k+1} := 1-e+\frac{c_k}{c},$$

and

$$d_1 := d,\ d_{k+1} := \frac{d}{c} \cdot c_k.$$

Suppose for the moment that $c_k \le c(1-e)/(c-1)$ for all k. By induction on k, it is easy to see that in this case, the sequences $\{c_k\}$ and $\{d_k\}$ are monotone nondecreasing: For $\{c_k\}$, we observe

$$c_{k+1} \ge c_k \iff 1-e+c_k/c \ge c_k \iff \frac{c(1-e)}{c-1} \ge c_k.$$

For $\{d_k\}$, we have

$$d_{k+1} \ge d_k \iff d \cdot c_{k+1} \ge cd_k = d \cdot c_k.$$

Since $c_1 \ge c$, we have $c_k \ge c$ for all k. Since the sequences are monotone nondecreasing, it follows that the sequence $\{c_k\}$ converges to

$$c_\infty = c(1-e)/(c-1).$$

(Supposing instead there is a k such that $c_k > c(1-e)/(c-1)$, then we also have $c_k \ge c$, and we can use that value of c_k in the below, which will only improve the argument.)

We will prove that, for all k,

$$\mathsf{DTISP}[n^{c_k}, n^e] \subseteq (\exists\, n^{1+o(1)})(\forall\, \log n)\mathsf{DTISP}[n^{1+o(1)}, n^{d+o(1)}]. \qquad (8)$$

and conclude that for all $c' < c_\infty = c(1-e)/(c-1)$,

$$\mathsf{DTISP}[n^{c'}, n^e] \subseteq (\exists\, n^{1+o(1)})(\forall\, \log n)\mathsf{DTISP}[n^{1+o(1)}, n^{d+o(1)}]. \tag{9}$$

By the Speedup Lemma,

$$\mathsf{DTISP}[n^{c_1}, n^{e+o(1)}] = \mathsf{DTISP}[n^{2-e-d}, n^{e+o(1)}] \subseteq (\exists\, n^{1+o(1)})(\forall\, \log n)\mathsf{DTISP}[n, n^{e+o(1)}].$$

For the inductive step, we have the following (subtle) series of inclusions:

$$\begin{aligned}
\mathsf{DTISP}[n^{c_{k+1}}, n^e] &\subseteq (\exists\, n^{1+o(1)})(\forall\, \log n)\mathsf{DTISP}[n^{c_{k+1}-(1-e)}, n^e] \quad \text{(Speedup)}\\
&= (\exists\, n^{1+o(1)})(\forall\, \log n)\mathsf{DTISP}[n^{c_k/c}, n^e] \quad \text{(def. of } c_{k+1}, \text{ \& } c_k \ge c)\\
&\subseteq (\exists\, n^{1+o(1)})\mathsf{DTISP}[n^{c_k+o(1)}, n^{d_k+o(1)}] \quad \text{(Slowdown \& def. of } d_k)\\
&\subseteq (\exists\, n^{1+o(1)})(\forall\, \log n)\mathsf{DTISP}[n^{1+o(1)}, n^{d+o(1)}] \quad \text{(by induction).}
\end{aligned}$$

For all sufficiently small $\varepsilon > 0$, let k be such that $c_k > c(1-e)/(c-1) - \varepsilon$. For sufficiently large time functions t, we have

$$\mathsf{DTISP}[t^{c(1-e)/(c-1)-\varepsilon}, t^{e+o(1)}] \subseteq (\exists\, t^{1+o(1)})(\forall\, \log t)\mathsf{DTISP}[t^{1+o(1)}, t^{d+o(1)}]$$

$$\subseteq \mathsf{NTIME}[t^{c+o(1)}] \subseteq \mathsf{DTISP}[t^{c^2+o(1)}, t^{cd}],$$

where the first inclusion follows from the assumption $\mathsf{NTIME}[n] \subseteq \mathsf{DTISP}[n^{c+o(1)}, n^{d+o(1)}]$, the second from (9), and the third from Slowdown. Therefore if c and d satisfy

$$c(1-e)/(c-1) - \varepsilon > c^2, \;\; e > cd$$

then we have a contradiction to the time-space hierarchy. Adding the two equations together, we obtain

$$e + \frac{c(1-e)}{c-1} - \varepsilon > c(c+d)$$

$$\implies \frac{e}{c} + \frac{1-e}{c-1} - \frac{\varepsilon}{c} > c+d.$$

Simplifying the LHS, we have

$$\frac{c-e}{c(c-1)} > c+d+O(\varepsilon),$$

and it is evident that the LHS is less than $1/(c-1)$ for $e > 0$. Therefore we have a contradiction when $(c+d)(c-1) < 1$. Recalling that ϕ is the smallest positive root of the equation $x(x-1) = 1$, we observe that $c+d < \phi$ implies

$$(c+d)(c-1) < (c+d)(c+d-1) < 1-\delta$$

for some fixed $\delta > 0$, contradicting the necessary condition. □

The above proof uses annotations of the form

$$[1,1,0,1,0,1,0,\ldots,1,0,0],$$

similar to the n^ϕ time and $n^{o(1)}$ space lower bound of Fortnow *et al.* [Fortnow et al. 2005]. However, the best lower bound proofs found by experiment actually have annotations analogous to that of the $2\cos(\pi/7)$ lower bound, of the form

$$[1,1,\ldots,1,1,0,1,0,\ldots,1,0,0,1,0,1,0,\ldots,1,0,0,\cdots,1,1,0,1,0,\ldots,1,0,0].$$

Unfortunately, the actual setting of parameters becomes vastly more complex, yielding systems of polynomial inequalities. As subsequent work has already improved these arguments (and the underlying automated lower bound framework), we will only provide the "next stage of induction" here, considering proofs with annotations of the form

$$[1,1,1,0,1,0,1,0,\ldots,1,0,0,1,0,1,0,1,0,\ldots,1,0,0].$$

To this end, we use the same strategy as in Theorem 4.11, letting $e > 0$ be a small parameter and defining the sequence $\{c_k\}$ just as before. From the proof of Theorem 4.11, we know we can assume that $c^2 \geq c_k$ for all k (otherwise, we obtain a contradiction). However, the kth derivation becomes:

$$\begin{aligned}
\mathsf{DTISP}[n^{c_k+(c^2/c_k)-e}, n^e] &\subseteq (\exists\, n^{c^2/c_k})(\forall\ \log n)\mathsf{DTISP}[n^{c_k}, n^e] \quad \text{(Speedup)}\\
&\subseteq (\exists\, n^{c^2/c_k})(\forall\ \log n)(\forall\ n)(\exists\ \log n)\mathsf{DTISP}[n^{1+o(1)}, n^e] \quad \text{(Speedup)}\\
&\subseteq (\exists\, n^{c^2/c_k})(\forall\ \log n)(\forall\ n)\mathsf{DTISP}[n^{c+o(1)}, n^d] \quad \text{(Slowdown)}\\
&\subseteq (\exists\, n^{c^2/c_k})\mathsf{DTISP}[n^{c^2+o(1)}, n^d] \quad \text{(Slowdown, \& } c \leq c_k)\\
&\subseteq (\exists\, n^{c^2/c_k})(\forall\ \log n)\mathsf{DTISP}[n^{c^2/c_k+o(1)}, n^{dc^2/c_k}] \quad \text{(Eq. (8))}\\
&\subseteq (\exists\, n^{c^2/c_k})\mathsf{DTISP}[n^{c^3/c_k+o(1)}, n^{dc^2/c_k}] \quad \text{(Slowdown)}\\
&\subseteq \mathsf{DTISP}[n^{c^4/c_k+o(1)}, n^{dc^3/c_k}] \quad \text{(Slowdown).}
\end{aligned}$$

A contradiction arises when c, d, e satisfy

$$e > dc^3/c_k,\ \ c_k + (c^2/c_k) - e > c^4/c_k.$$

As argued in the proof of Theorem 4.11, we may assume that $c_k = (1-e)c/(c-1) - \varepsilon$ for arbitrarily small $\varepsilon > 0$. This substitution reduces the system of polynomial inequalities to:

$$dc^3(c-1) + O(\varepsilon) > ec(1-e),\ \ c^4(c-1)^2 + O(\varepsilon) > 2c^2 - 3c^2e + 2c^2e^2 + c^4 - 2c^3 + ce - ce^2.$$

However, this is still not enough to tractably analyze the tradeoff between c and d. The following table arises from setting various values of d, then analytically solving to maximize c such that the above system is satisfied, for some e. (The precise command executed in Maple was:

```
with(Optimization); d := x;
 Maximize(c, {e >= d*c^3/(c*(1-e)/(c-1)-0.1e-5),
 c*(1-e)/(c-1)-0.1e-5 + c^2/(c*(1-e)/(c-1)-0.1e-5) - e
 >= c^4/(c*(1-e)/(c-1)-0.1e-5), c^2 >= c*(1-e)/(c-1),
 e <= 2-c, c+d <= 2}, assume = nonnegative, initialpoint = [c = 1.7, e = 0]);
```

where `x` denotes the desired space bound.)

Time	Space
$n^{1.05}$	$n^{.9}$
$n^{1.13}$	$n^{.75}$
$n^{1.18}$	$n^{.666}$
$n^{1.28}$	$n^{.5}$
$n^{1.38}$	$n^{.333}$
$n^{1.46}$	$n^{.25}$
$n^{1.60}$	$n^{.1}$
$n^{1.66}$	$n^{.05}$

As one can see, these values bring us slightly closer to those of Table I than the guarantee of $c + d < 1.618$ from Theorem 4.11. More involved proof annotations (with much more complicated polynomial inequalities) yield even better tables, culminating in the time-space tradeoffs found by computer, described by the author and Buss [Buss and Williams 2012].

5. TIME-SPACE LOWER BOUNDS FOR SOLVING QUANTIFIED BOOLEAN FORMULAS

We can generalize the lower bounds of the previous section to quantified Boolean formulas with a fixed number of quantifier blocks (i.e., alternations). As first observed by Fortnow and Van Melkebeek [Fortnow and van Melkebeek 2000], the alternation-trading scheme for lower bounds against nondeterminism extends naturally to lower bounds against alternating computations. Since $\mathsf{AP} = \mathsf{PSPACE}$ [Chandra et al. 1981], it follows that $\mathsf{ATIME}[n] \nsubseteq \mathsf{DTISP}[n^k, n^{o(1)}]$ for *every* $k \geq 1$.[10] So we already have a polynomial time lower bound for the most general version of the quantified Boolean formula problem (QBF) in the small space setting.

How large can lower bounds for quantified Boolean formulas get, when the number of quantifier blocks is a fixed constant? Define QBF_k to be the problem of solving a QBF with k quantifier blocks (*i.e.* deciding the truth of Σ_k and Π_k sentences in first-order Boolean logic). Building on Fortnow and Van Melkebeek [Fortnow and van Melkebeek 2000] who proved that QBF_k cannot be solved in $O(n^{k-\varepsilon})$ time on $n^{o(1)}$-space machines, we prove time lower bounds for QBF_k of the form $n^{k+1-\varepsilon_k}$ on the same model, where $\{\varepsilon_k\}$ is a decreasing sequence such that $\lim_{k\to\infty} \varepsilon_k = 0$.

THEOREM 5.1. *For all $k \geq 1$, QBF_k cannot be solved in $O(n^c)$ time on $n^{o(1)}$ space RAMs, for all c satisfying $c^3/k - c^2 - 2c + k < 0$.*

Note this result generalizes the $2\cos(\pi/7)$ lower bound for SAT. The remainder of this section proves Theorem 5.1, which was inspired by some short proofs generated by our theorem prover. As our argument closely follows that of other proofs in the paper, we keep the exposition at a more informal level.

We use the fact that QBF_k is "robustly complete" in the appropriate sense, then show $\Sigma_k\mathsf{TIME}[n] \nsubseteq \mathsf{DTS}[n^c]$ for certain $c > k$ by proving a series of class containments. Let us recall the completness result of Fortnow *et al.*, which follows from Theorem 3.1:

THEOREM 5.2 (FORTNOW-LIPTON-VAN MELKEBEEK-VIGLAS [FORTNOW ET AL. 2005]). *For all $k \geq 1$, QBF_k is robustly complete for $\Sigma_k\mathsf{QL} \cup \Pi_k\mathsf{QL}$. In particular, there is a quasi-linear reduction from an arbitrary language in the class to QBF_k, where an arbitrary bit of the reduction can be computed in polylogarithmic time.*

We can modify the LP framework for SAT lower bounds to obtain a similar LP framework for QBF_k lower bounds: only the Slowdown Rule differs, as its application removes k quantifiers instead of just one. Doing so, we wrote a program for QBF_k time lower bounds, which produced proofs that inspired the below development. (We omit the description of the linear programming formulation here, as it is quite similar to that of the previous section.)

The main tool we use is the following.

THEOREM 5.3 (CONDITIONAL SPEEDUP FOR THE POLYNOMIAL-TIME HIERARCHY). *If $\Sigma_k\mathsf{TIME}[n] \subseteq \mathsf{DTS}[n^c]$ for some c satisfying $k < c < k+1$, then for all d satisfying $c \leq d < \frac{c}{c-k}$, $\mathsf{DTS}[n^d] \subseteq \Sigma_{k+1}\mathsf{TIME}[n^{1+o(1)}] \cap \Pi_{k+1}\mathsf{TIME}[n^{1+o(1)}]$.*

PROOF. Similar to the proof of the Conditional Speedup Theorem in [Williams 2008]. We show that $\mathsf{DTS}[n^d] \subseteq \Sigma_{k+1}\mathsf{TIME}[n^{1+o(1)}] \cap \Pi_{k+1}\mathsf{TIME}[n^{1+o(1)}]$ implies $\mathsf{DTS}[n^{1+dk/c}] \subseteq \Sigma_{k+1}\mathsf{TIME}[n^{1+o(1)}] \cap \Pi_{k+1}\mathsf{TIME}[n^{1+o(1)}]$. This process converges when $d = 1 + dk/c$, or $d = c/(c-k)$.

The Speedup Lemma (Lemma 3.3) implies that

$$\mathsf{DTS}[n^{1+dk/c}] \subseteq (\exists\, n)(\forall\, \log n)\mathsf{DTS}[n^{dk/c}].$$

[10]Otherwise, $\mathsf{SPACE}[n] \subseteq \mathsf{ATIME}[n^2] \subseteq \mathsf{SPACE}[n^{o(1)}]$, contradicting the space hierarchy theorem.

Applying speedup for $k-1$ times,

$$\mathsf{DTS}[n^{1+dk/c}] \subseteq (\exists\ n)(\forall\ \log n)\underbrace{(\forall\ n^{d/c})\cdots(Q\ n^{d/c})}_{k-1}(\neg Q\ \log n)\mathsf{DTS}[n^{d/c}]$$

for some $Q \in \{\exists, \forall\}$, where $\neg Q$ is opposite to Q. Since $\Sigma_k\mathsf{TIME}[n] \subseteq \mathsf{DTS}[n^c]$,

$$(\exists\ n)\underbrace{(\forall\ n^{d/c})\cdots(Q\ n^{d/c})(\neg Q\ \log n)}_{k}\mathsf{DTS}[n^{d/c}] \subseteq (\exists\ n)\mathsf{DTS}[n^d].$$

Finally, since $\mathsf{DTS}[n^d] \subseteq \Sigma_{k+1}\mathsf{TIME}[n^{1+o(1)}] \cap \Pi_{k+1}\mathsf{TIME}[n^{1+o(1)}]$,

$$(\exists\ n)\mathsf{DTS}[n^d] \subseteq \Sigma_{k+1}\mathsf{TIME}[n^{1+o(1)}].$$

An analogous argument implies $\mathsf{DTS}[n^{1+dk/c}] \subseteq \Pi_{k+1}\mathsf{TIME}[n^{1+o(1)}]$. □

THEOREM 5.4. *If* $\Sigma_k\mathsf{TIME}[n] \subseteq \mathsf{DTS}[n^c]$*, then for all* $\ell \geq 1$ *and* d *satisfying* $c \leq d < c/(c-k)$*,*

$$\mathsf{DTS}[n^{d+\sum_{i=1}^{\ell}\left(\frac{c^2}{dk}\right)^i}] \subseteq \Sigma_{k+1}\mathsf{TIME}[n^{\left(\frac{c^2}{dk}\right)^\ell+o(1)}] \cap \Pi_{k+1}\mathsf{TIME}[n^{\left(\frac{c^2}{dk}\right)^\ell+o(1)}].$$

PROOF. Induction on ℓ. The case $\ell = 0$ is immediate, by the previous theorem. For the inductive step, suppose $\mathsf{DTS}[n^{d+\sum_{i=1}^{\ell}\left(\frac{c^2}{dk}\right)^i}] \subseteq \Sigma_{k+1}\mathsf{TIME}[n^{\left(\frac{c^2}{dk}\right)^\ell+o(1)}]$. First, the Speedup Lemma implies

$$\mathsf{DTS}[n^{d+\sum_{i=1}^{\ell+1}\left(\frac{c^2}{dk}\right)^i}] \subseteq (\exists\ n^{\left(\frac{c^2}{dk}\right)^{\ell+1}})(\forall\ \log n)\mathsf{DTS}[n^{d+\sum_{i=1}^{\ell}\left(\frac{c^2}{dk}\right)^i}],$$

where the input to the DTS part has length $n + 2n^{o(1)}$. By the induction hypothesis, the above is contained in

$$(\exists\ n^{\left(\frac{c^2}{dk}\right)^{\ell+1}})(\forall\ \log n)\Pi_{k+1}\mathsf{TIME}[n^{\left(\frac{c^2}{dk}\right)^\ell+o(1)}].$$

Applying $\Sigma_k\mathsf{TIME}[n] \subseteq \mathsf{DTS}[n^c]$ to the Σ_k part of the $\Pi_{k+1}\mathsf{TIME}$ class, the above class is contained in

$$\exists\ n^{\left(\frac{c^2}{dk}\right)^{\ell+1}})(\forall\ \log n)(\forall\ n^{\left(\frac{c^2}{dk}\right)^\ell})\mathsf{DTS}[n^{c\left(\frac{c^2}{dk}\right)^\ell}].$$

If $c < k$, then there is no d satisfying $c \leq d < c/(c-k)$, so there is nothing to prove. If $c \geq k$, the above class is contained in

$$(\exists\ n^{\left(\frac{c^2}{dk}\right)^{\ell+1}})(\forall\ \log n)(\forall\ n^{\left(\frac{c^2}{dk}\right)^\ell})\Pi_k\mathsf{TIME}[n^{\frac{c}{k}\cdot\left(\frac{c^2}{dk}\right)^\ell+o(1)}] = (\exists\ n^{\left(\frac{c^2}{dk}\right)^{\ell+1}})\Pi_k\mathsf{TIME}[n^{\frac{c}{k}\cdot\left(\frac{c^2}{dk}\right)^\ell+o(1)}].$$

Note $(\frac{c^2}{dk})^{\ell+1} \leq \frac{c}{k}\cdot(\frac{c^2}{dk})^\ell$, because $d \geq c$. Invoking the assumption $\Sigma_k\mathsf{TIME}[n] \subseteq \mathsf{DTS}[n^c]$ again results in the class

$$(\exists\ n^{\left(\frac{c^2}{dk}\right)^{\ell+1}})\mathsf{DTS}[n^{\frac{c^2}{k}\cdot\left(\frac{c^2}{dk}\right)^\ell}].$$

Finally, since $d(\frac{c^2}{dk})^{\ell+1} = \frac{c^2}{k}\cdot(\frac{c^2}{dk})^\ell$, Theorem 5.3 applies, and the above class is in

$$(\exists\ n^{\left(\frac{c^2}{dk}\right)^{\ell+1}})\Sigma_{k+1}\mathsf{TIME}[n^{\frac{c^2}{dk}\cdot\left(\frac{c^2}{dk}\right)^\ell+o(1)}] = \Sigma_{k+1}\mathsf{TIME}[n^{\left(\frac{c^2}{dk}\right)^{\ell+1}+o(1)}].$$

An analogous argument proves the containment for $\Pi_{k+1}\mathsf{TIME}[n^{\left(\frac{c^2}{dk}\right)^{\ell+1}+o(1)}]$. □

Finally, we prove Theorem 5.1. Let $K_\ell = d + \sum_{i=1}^{\ell}\left(\frac{c^2}{dk}\right)^i$, for $\ell \geq 1$. We claim (the proof is not hard) that

$$\left(\frac{c^2}{dk}\right)^\ell \leq K_\ell\left(1 - \frac{dk}{c^2} - \varepsilon_\ell\right),$$

Problem	Time Lower Bound Exponent
SAT	$n^{1.801}$
QBF$_2$	$n^{2.903}$
QBF$_3$	$n^{3.942}$
QBF$_4$	$n^{4.962}$
QBF$_{10}$	$n^{10.991}$
QBF$_{100}$	$n^{100.999902}$

Table II.

for a small constant $\varepsilon_\ell > 0$ satisfying $\lim_{\ell\to\infty} \varepsilon_\ell = 0$. Applying the above theorems, we deduce the chain:

$$\begin{aligned}\Sigma_{k+1}\mathsf{TIME}[n^{K_\ell}] &\subseteq (\exists\, n^{K_\ell})\mathsf{DTS}[n^{cK_\ell}] \\ &\subseteq (\exists\, n^{K_\ell})\Sigma_k\mathsf{TIME}[n^{(c/k)K_\ell}] \subseteq \mathsf{DTS}[n^{(c^2/k)K_\ell}] \subseteq \Pi_{k+1}\mathsf{TIME}[n^{(c^2/k)K_\ell\left(1-\frac{dk}{c^2}-\varepsilon_\ell\right)}].\end{aligned}$$

For sufficiently large ℓ, a contradiction is reached when

$$c^2/k(1-(dk)/(c^2)) < 1, \tag{10}$$

i.e., when $c^2/k - d - 1 < 0$. Recalling that $d < c/(c-k)$, we let $d = c/(c-k) - \varepsilon$ for an arbitrarily small $\varepsilon > 0$. Then (10) becomes $c^2/k - c/(c-k) + \varepsilon - 1 < 0$. Rearranging, we reach a contradiction when

$$c^3/k - c^2 - 2c + k + \varepsilon(c-k) < 0. \tag{11}$$

For any c such that $c^3/k - c^2 - 2c + k < 0$, we can choose ε small enough that condition (11) still holds, and a contradiction is reached.

Table II summarizes the concrete lower bounds. As the evidence suggests, at least one root of the polynomial $p_k(c) = c^3/k - c^2 - 2c + k$ gradually approaches $k+1$ as k increases unboundedly; hence the lower bound exponent for QBF$_k$ approaches $k+1$.

PROPOSITION 5.5. $\lim_{k\to\infty} p_k(k+1) = 0$. *In particular, for all k,* $p_k(k+1-1/k) < 0$ *and* $p_k(k+1) > 0$.

PROOF. Algebraic manipulation gives $p_k(k+1) = 1/k > 0$ and $p_k(k+1-1/k) = 3/k^3 - 1 - 1/k - 1/k^2 - 1/k^4 < -1/k^4 < 0$, for all $k \geq 1$. □

6. NONDETERMINISTIC TIME-SPACE LOWER BOUNDS FOR TAUTOLOGIES

The problem of proving nondeterministic time-space lower bounds for coNP has also been studied in the literature. Fortnow and Van Melkebeek [Fortnow and van Melkebeek 2000] proved that TAUTOLOGY cannot be solved in $n^{\sqrt{2}-o(1)}$ time on a nondeterministic $n^{o(1)}$ space RAM. However, since their initial result, no further improvements had been found. We show how to extend the approach of the previous section to this problem, and find that the best proof annotations look quite different. Here the approach turns out to be successful in finding new proofs.

6.1. The Framework and Linear Programming Translation

Similar to the class DTS, set $\mathsf{NTS}[n^a] := \mathsf{NTISP}[n^a, n^{o(1)}]$ and $\mathsf{coNTS}[n^a] := \mathsf{coNTISP}[n^a, n^{o(1)}]$ for brevity. As in Section 3, there are Speedup and Slowdown rules that are applied in some way that contradicts a time hierarchy, although the rules are somewhat different here. In the following, let Q be a string of quantifier blocks, so $Q = (Q_1\, n^{a_1})^{b_2} \cdots (Q_{k-1}\, n^{a_{k-1}})$.

LEMMA 6.1. *(Speedup) For* $b \geq 1$, $a \geq 1$, $x \geq 0$, *and* $s \geq 0$,

$$Q^b\mathsf{NTISP}[n^a, n^s] \subseteq Q^b(\exists\, n^{x+s})^{\max\{b,x+s\}}(\forall\, \log n)^{\max\{b,s\}}\mathsf{NTISP}[n^{a-x}, n^s].$$

In particular for $s = o(1)$ we have $\mathsf{NTS}[n^a] \subseteq (\exists\, n^x)^{\max\{1,x\}}(\forall\, \log n)^1 \mathsf{NTS}[n^{a-x}]$.

PROOF. The proof is analogous to Lemma 3.3 (the Speedup Lemma for DTISP). □

LEMMA 6.2. *(Slowdown) If* TAUTOLOGY *is in* $\mathsf{NTS}[n^c]$ *then*

(1) $Q^b(\exists\, n^{a_k})^{b_{k+1}}\mathsf{NTS}[n^{a_{k+1}}] \subseteq Q^b\mathsf{coNTS}[n^{c\cdot\max\{a_k,a_{k+1},b_{k+1},b\}}]$,
(2) $Q^b(\forall\, n^{a_k})^{b_{k+1}}\mathsf{coNTS}[n^{a_{k+1}}] \subseteq Q^b\mathsf{NTS}[n^{c\cdot\max\{a_k,a_{k+1},b_{k+1},b\}}]$,
(3) $Q^b(\exists\, n^{a_k})^{b_{k+1}}\mathsf{coNTS}[n^{a_{k+1}}] \subseteq Q^b(\exists\, n^{a_k})^{b_{k+1}}\mathsf{NTS}[n^{c\cdot\max\{a_{k+1},b_{k+1}\}}]$, *and*
(4) $Q^b(\forall\, n^{a_k})^{b_{k+1}}\mathsf{NTS}[n^{a_{k+1}}] \subseteq Q^b(\forall\, n^{a_k})^{b_{k+1}}\mathsf{coNTS}[n^{c\cdot\max\{a_{k+1},b_{k+1}\}}]$.

The proofs are omitted, as they follow analogously to the Slowdown Lemma for DTISP. To obtain contradictions, one uses the alternating time hierarchy (Theorem 3.5) just as in the deterministic case. The above lemmas immediately lead to a natural definition of *alternation-trading proof that* $\mathsf{coNTIME}[n] \subseteq \mathsf{NTS}[n^c] \Longrightarrow A_1 \subseteq A_2$, for classes A_1 and A_2. Similar to Lemma 4.3, which showed that $\mathsf{NTIME}[n] \subseteq \mathsf{DTS}[n^c]$ implies $\mathsf{DTS}[n^a] \not\subseteq \mathsf{DTS}[n^{a'}]$ for $a > a'$, we can prove:

LEMMA 6.3. *If* $\mathsf{coNTIME}[n] \subseteq \mathsf{NTS}[n^c]$ *for some c, then* $\mathsf{NTS}[n^a] \not\subseteq \mathsf{coNTS}[n^{a'}]$ *for all* $a > a' \geq 1$.

The proof is completely analogous to the first case of the proof of Lemma 4.3.

PROOF. Suppose that $\mathsf{coNTIME}[n] \subseteq \mathsf{NTS}[n^c]$ and $\mathsf{NTS}[n^a] \subseteq \mathsf{coNTS}[n^{a'}]$ for some $a > a' \geq 1$. We wish to prove a contradiction. Let $\delta = a - a'$, so that $\mathsf{NTS}[n^a] \subseteq \mathsf{coNTS}[n^{a-\delta}]$. By complementing machines, we also have $\mathsf{coNTS}[n^a] \subseteq \mathsf{NTS}[n^{a-\delta}]$. By a translation/padding argument, $\mathsf{NTS}[n^a] \subseteq \mathsf{coNTS}[n^{a-\delta}]$ implies

$$\mathsf{NTS}[n^{a^2/(a-\delta)}] \subseteq \mathsf{coNTS}[n^a] \subseteq \mathsf{NTS}[n^{a-\delta}]$$

and $\mathsf{coNTS}[n^{a^2/(a-\delta)}] \subseteq \mathsf{coNTS}[n^{a-\delta}]$ as well. Repeating this padding for an arbitrary finite number of times, we obtain

$$\mathsf{NTS}[n^{a\cdot(a/(a-\delta))^i}] \subseteq \mathsf{coNTS}[n^{a-\delta}] \text{ for all even } i \geq 0, \text{ and}$$

$$\mathsf{NTS}[n^{a\cdot(a/(a-\delta))^i}] \subseteq \mathsf{NTS}[n^{a-\delta}] \text{ for all odd } i \geq 1.$$

Since $\delta > 0$, this implies $\mathsf{NTS}[n^L] \subseteq \mathsf{NTS}[n^{a-\delta}]$ for all odd $L \geq a - \delta$. Therefore, if $\mathsf{coNTIME}[n] \subseteq \mathsf{NTS}[n^c]$ then for all odd $L \geq a$,

$$\mathsf{coNTIME}[n^L] \subseteq \mathsf{NTS}[n^{Lc}] \subseteq \mathsf{NTS}[n^{a-\delta}] \subseteq \mathsf{NTIME}[n^{a-\delta}],$$

a contradiction to the time hierarchy (Theorem 3.5). □

Lemma 6.3 motivates a definition of *normal form proof* for $\mathsf{coNTIME}[n] \not\subseteq \mathsf{NTS}[n^c]$, and in an analogous way we can prove that any alternation-trading proof can be converted into normal form.

Definition 6.4. Let $c \geq 1$. An alternation-trading proof that ($\mathsf{coNTIME}[n] \subseteq \mathsf{NTS}[n^c] \Longrightarrow A_1 \subseteq A_2$) is in *normal form* if *(1)* $A_1 = \mathsf{NTS}[n^a]$, $A_2 = \mathsf{coNTS}[n^{a'}]$, for some $a \geq a'$, and *(2)* no other lines are NTS or coNTS classes.

Example. To get a feeling for how these ingredients interact, let us consider a small example proof. If TAUTOLOGY $\in \mathsf{NTS}[n^c]$ then $\mathsf{coNTIME}[n] \subseteq \mathsf{NTS}[n^{c+o(1)}]$ by Theorem 3.1, so

$$\mathsf{NTS}[n^2] \subseteq (\exists\, n)^1(\forall\, \log n)^1\mathsf{NTS}[n]$$

by Lemma 6.1 (NTISP Speedup). Applying Lemma 6.2 (NTISP Slowdown) thrice,

$$(\exists\, n)^1(\forall\, \log n)^1\mathsf{NTS}[n] \subseteq (\exists\, n)^1(\forall\, \log n)^1\mathsf{coNTS}[n^c] \subseteq (\exists\, n)^1\mathsf{NTS}[n^{c^2}] \subseteq \mathsf{coNTS}[n^{c^3}].$$

When $c < \sqrt[3]{2} \approx 1.25$, we derive $\mathsf{NTS}[n^a] \subseteq \mathsf{coNTS}[n^{a'}]$ for some $a > a'$, which contradicts Lemma 6.3. Note that this lower bound looks rather similar to the $\sqrt{2}$ lower bound from Section 3, but we must apply three slowdowns rather than just two.

LEMMA 6.5. *Let $c \geq 1$. If there is an alternation-trading proof that* $(\mathsf{coNTIME}[n] \subseteq \mathsf{NTS}[n^c] \Longrightarrow A_1 \subseteq A_2)$ *in normal form, and the proof has at least two lines, then* $\mathsf{coNTIME}[n] \not\subseteq \mathsf{NTS}[n^c]$.

PROOF. (Sketch) We follow the proof of Lemma 4.3 very closely. Let $\mathsf{NTS}[n^a]$ and $\mathsf{coNTS}[n^{a'}]$ be the first and last classes of a normal-form proof. If $a > a'$, then Lemma 6.3 already applies (analogously to the first case of Lemma 4.3). If $a = a'$, then $\mathsf{NTS}[n^a] = \mathsf{coNTS}[n^a]$, and we argue analogously to the second case of Theorem 4.3: there must be some class A in the normal form proof such that A contains a positive number of alternations (guessing at least n^δ bits for some $\delta > 0$), $A = \mathsf{NTS}[n^a] = \mathsf{coNTS}[n^a] = A'$, where A' is the complementary class to A, and we have $A' = (\forall\, n^\delta)A'$ and $A = (\exists\, n^\delta)A$. By Lemma 6.1, we have for every k that

$$\mathsf{NTS}[n^k] \subseteq \mathsf{NTS}[n^{\delta\lceil \frac{k}{\delta}\rceil}] \subseteq \underbrace{(\exists\, n^\delta)(\forall\, n^\delta)\cdots(\exists\, n^\delta)(\forall\, n^\delta)}_{O(k/\delta)}A'.$$

But the RHS is just the class A, as argued in Lemma 4.3. Therefore by padding,

$$\mathsf{NTIME}[n^{a+1}] \subseteq \mathsf{coNTS}[n^{(a+1)c}] = \mathsf{NTS}[n^{(a+1)c}] \subseteq A = \mathsf{coNTS}[n^a] \subseteq \mathsf{coNTIME}[n^a]$$

which contradicts the time hierarchy theorem. $\square$

THEOREM 6.6. *Let A_1 and A_2 be simple and complementary. If there is an alternation-trading proof P that* $(\mathsf{coNTIME}[n] \subseteq \mathsf{NTS}[n^c] \Longrightarrow A_1 \subseteq A_2)$*, then there is a normal form proof for c, of length at most that of P.*

The proof of Theorem 6.6 is analogous to that of Theorem 4.4. One can also define proof annotations for this setting. The vectors corresponding to valid annotations change due to differences in the rules. For example, note $[1,0,0,0]$, $[1,1,0,0,0,0,0]$, and $[1,0,1,0,0,0,0]$ are valid annotations for this setting, the first being the annotation for the above example. From the Speedup and Slowdown Lemmas given above, observe that the operations on exponents are again max, $+$, and multiplication by c. Hence the translation of annotations to linear programming follows a similar strategy as before: we define variables $a_{i,j}$, $b_{i,j}$, x_i for all lines i and possible quantifier blocks j, replace components of the form $\max\{a,a'\} = a''$ with $a'' \geq a$, $a'' \geq a'$, then minimize $\sum a_{i,j} + b_{i,j} + x_i$.

6.2. Empirical Results

Although the proof rules used in the "TAUTOLOGY versus NTISP" lower bound problem look extremely similar to those in the "SAT versus DTISP" problem, the structure of good lower bound proofs for "TAUTOLOGY versus NTISP" turns out to be quite different. Below is a table of results found by exhaustive search over valid annotations.[11]

[11]In this lower bound setting, there are no valid proof annotations of length 3, 4, 6, 7, 9, 10, 12, 13, *etc.* This is tedious to prove, so we omit it.

k	Optimal Proof Annotation(s) With k Lines	Time L.B. Exponent
5	[1, 0, 0, 0]	1.323
8	[1, 1, 0, 0, 0, 0, 0]	1.380
	[1, 0, 1, 0, 0, 0, 0]	
11	[1, 1, 0, 0, 0, 1, 0, 0, 0, 0]	1.419
14	[1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0]	1.433
	[1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0]	
	[1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0]	
	[1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0]	
17	[1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0]	1.445
	[1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0]	
20	[1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0]	1.455
	[1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0]	
	[1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0]	
	[1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0]	
23	[1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0]	1.465

For example, the best 11-line proof is:

```
0, NTS[n^4.788956584]
1, (E n^2.369956583)(A n^1.)NTS[n^2.419]
2, (E n^2.369956583)(A n^1.)(E n^1.419)(A n^1.)NTS[n^1.]
3, (E n^2.369956583)(A n^1.)(E n^1.419)(A n^1.)coNTS[n^1.419]
4, (E n^2.369956583)(A n^1.)(E n^1.419)NTS[n^2.013561000]
5, (E n^2.369956583)(A n^1.)coNTS[n^2.857243059]
6, (E n^2.369956583)(A n^2.378351877)(E n^1.)coNTS[n^1.181167036]
7, (E n^2.369956583)(A n^2.378351877)(E n^1.)NTS[n^1.676076023]
8, (E n^2.369956583)(A n^2.378351877)coNTS[n^2.378351877]
9, (E n^2.369956583)NTS[n^3.374881314]
10, coNTS[n^4.788956584]
```

Notice how larger annotations are composed of smaller ones: for example, $[1,1,1,0,0,0,0,0,1,0,0,0,0]$ is $[1,A_7,A_4,0]$, where A_4 and A_7 are optimal annotations for four and seven lines.[12] In particular, observe that three optimal annotations from the table have a distinctive pattern, namely

$$A=[1,0,0,0], \hat{A}=[1,1,0,0,0,1,0,0,0,0], \text{ and } \hat{\hat{A}}=[1,1,1,0,0,0,1,0,0,0,0,1,1,0,0,0,1,0,0,0,0,0].$$

The key observation is that $\hat{A}=[1,A,A,0]$ and $\hat{\hat{A}}=[1,\hat{A},\hat{A},0]$. This pattern suggests that we look for a proof by induction, where we start with a speedup, apply an induction hypothesis twice, then apply a slowdown. For example, the next annotation in the pattern would be

$$\begin{aligned}[1,\hat{\hat{A}},\hat{\hat{A}},0] \;=\; & [1,1,1,1,0,0,0,1,0,0,0,0,1,1,0,0,0,1,0,0,0,0,0,\\ & 1,1,1,0,0,0,1,0,0,0,0,1,1,0,0,0,1,0,0,0,0,0,0],\end{aligned}$$

a 47-line annotation that gives a 1.49 exponent. A heuristic search found the above 47-line annotation, and no other annotations found (with at most 47 lines) attained a lower bound of that quality. For many line numbers ℓ, heuristic search found a large number of ℓ-line proof annotations that achieve the same lower bound. For example, there are eight such annotations of 26 lines. Each optimal annotation found could be written as a concatenation of smaller optimal annotations along with an additional 1 and 0.

[12]Of course, we are slightly abusing notation here, as A_4 and A_7 are themselves vectors. When v is a k-bit vector and we write (for example) $w=[0,v,1]$, we mean that $w=[0,v[1],\ldots,v[k],1]$.

6.3. Interpreting The Results

Generalizing from the results above, we arrive at the following result (joint with Diehl and Van Melkebeek):

THEOREM 6.7 ([DIEHL ET AL. 2011]). TAUTOLOGY *requires* $n^{\sqrt[3]{4}-o(1)}$ *time for nondeterministic algorithms using* $n^{o(1)}$ *space.*

Our proof here differs from the treatment in [Diehl et al. 2011], in that we closely mimic the proofs generated by the theorem prover. For a given constant $c \geq 1$, define the sequence $x_1 := 1$, $x_2 := c$, $x_k := c^3(x_{k-1})^2/(\sum_{i=1}^{k-1} x_i)$.

Proof of Theorem 6.7. Suppose $\mathsf{coNTIME}[n] \subseteq \mathsf{NTS}[n^c]$. We will prove that $c < 4^{1/3}$ entails a contradiction. We make the following claim:

CLAIM 1. *For all* $k \geq 2$, $c^2 x_k \geq \sum_{i=1}^{k} x_i$ *and* $\mathsf{NTS}[n^{\sum_{i=1}^{k} x_i}] \subseteq (\exists\, n^{x_k})(\forall\, n^{x_k})\mathsf{coNTS}[n^{x_k}]$.

Our proof of Claim 1 is by induction on k. We can think of the proof as proceeding in stages, starting with stage $k = 2$. At the beginning of stage k, we may assume that $c^2 x_\ell \geq \sum_{i=1}^{\ell} x_i$ for all $\ell < k$, and that

$$\mathsf{NTS}[n^{\sum_{i=1}^{\ell} x_i}] \subseteq (\exists\, n^{x_\ell})(\forall\, n^{x_\ell})\mathsf{coNTS}[n^{x_\ell}].$$

(Note that $c^2 x_1 \geq x_1$, so these assumptions are valid at stage 2.) At the end of stage k, we either derive a contradiction to the assumption $\mathsf{coNTIME}[n] \subseteq \mathsf{NTS}[n^c]$, or conclude that $c^2 x_k \geq \sum_{i=1}^{k} x_i$ and

$$\mathsf{NTS}[n^{\sum_{i=1}^{k} x_i}] \subseteq (\exists\, n^{x_k})(\forall\, n^{x_k})\mathsf{coNTS}[n^{x_k}].$$

When $k = 2$, we can already derive

$$\mathsf{NTS}[n^{c+1}] \subseteq (\exists\, n^c)(\forall\, \log n)\mathsf{NTS}[n] \subseteq (\exists\, n^c)(\forall\, \log n)\mathsf{coNTS}[n^c].$$

So we can derive (via Lemma 6.2) the inclusion

$$\mathsf{NTS}[n^{c+1}] \subseteq (\exists\, n^c)(\forall\, \log n)\mathsf{coNTS}[n^c] \subseteq (\exists\, n^c)\mathsf{NTS}[n^{c^2}] \subseteq \mathsf{coNTS}[n^{c^3}],$$

which contradicts Lemma 6.3 if $c^3 = c^2 x_2 < x_1 + x_2 = 1 + c$. (Note that $c^3 \geq 1 + c$ implies $c \geq 1.324$.) Hence $c^2 x_2 \geq x_1 + x_2$.

Observe that $c^2 x_k \geq \sum_{i=1}^{k} x_i$ implies that $x_{k+1} \geq 1$. (We may assume by induction that $x_k \geq 1$.) At the beginning of stage $k+1$ we have $c^2 x_k \geq \sum_{i=1}^{k} x_i$, and we derive the sequence of inclusions

$$\begin{aligned}
\mathsf{NTS}[n^{\sum_{i=1}^{k+1} x_i}] &\subseteq (\exists\, n^{x_{k+1}})(\forall\, \log n)^1\ \mathsf{NTS}[n^{\sum_{i=1}^{k} x_i}] \text{ (Speedup)}\\
&\subseteq (\exists\, n^{x_{k+1}})(\forall\, \log n)^1(\exists\, n^{x_k})(\forall\, n^{x_k})\mathsf{coNTS}[n^{x_k}] \text{ (by Induction Hypothesis / Stage } k)\\
&\subseteq (\exists\, n^{x_{k+1}})(\forall\, \log n)^1(\exists\, n^{x_k})\mathsf{NTS}[n^{c x_k}] \text{ (Slowdown)}\\
&\subseteq (\exists\, n^{x_{k+1}})(\forall\, \log n)^1\mathsf{coNTS}[n^{c^2 x_k}] \text{ (Slowdown)}\\
&\subseteq (\exists\, n^{x_{k+1}})(\forall\, \log n)^1(\forall\, n^{c^2(x_k)^2/(\sum_i x_i)})(\exists\, n^{c^2(x_k)^2/(\sum_i x_i)})\mathsf{NTS}[n^{c^2(x_k)^2/(\sum_i x_i)}]\\
&\qquad\qquad \text{(by Induction Hypothesis / Stage } k\text{, and } c^2 x_k \geq \sum_{i=1}^{k} x_i)\\
&\subseteq (\exists\, n^{x_{k+1}})(\forall\, \log n)^1(\forall\, n^{c^2(x_k)^2/(\sum_i x_i)})\mathsf{coNTS}[n^{c^3(x_k)^2/(\sum_i x_i)}] \text{ (Slowdown)}\\
&\subseteq (\exists\, n^{x_{k+1}})(\forall\, n^{x_{k+1}})\mathsf{coNTS}[n^{x_{k+1}}].
\end{aligned}$$

(Note that this induction indeed applies the induction hypothesis twice, as suggested by experiments.)

At this point, if $c^2 x_{k+1} < \sum_{i=1}^{k+1} x_i$, then we may derive

$$\mathsf{NTS}[n^{\sum_{i=1}^{k+1} x_i}] \subseteq (\exists\, n^{x_{k+1}})(\forall\, n^{x_{k+1}})\mathsf{coNTS}[n^{x_{k+1}}] \subseteq (\exists\, n^{x_{k+1}})(\forall\, n^{x_{k+1}})\mathsf{coNTS}[n^{x_{k+1}}] \subseteq \mathsf{coNTS}[n^{c^2 x_{k+1}}]. \tag{12}$$

Therefore $c^2 x_k < \sum_{i=1}^{k} x_i$ implies that $\mathsf{NTS}[n^{a+\delta}] \subseteq \mathsf{coNTS}[n^a]$ for some $a \geq 1$ and $\delta > 0$, which is a contradiction by Lemma 6.3. This concludes stage k, and the proof of Claim 1.

Now define the sequence $s_k := (\sum_{i=1}^{k} x_i)/x_k = 1 + \sum_{i=1}^{k-1} x_i/x_k$. Because $c^2 x_k \geq \sum_{i=1}^{k} x_i$ for all k, we must have $c^2 \geq s_k$ for all k.

CLAIM 2. *For all $k > 2$, $s_k = 1 + (s_{k-1})^2/c^3$, and $\{s_k\}$ is increasing when $c < 4^{1/3}$.*

The proof is by induction on k. When $k = 3$, $s_2 = 1 + 1/c$ and $s_3 = 1 + (1+c)/c^5 + (1+c)/c^4 = 1 + (1+1/c)^2/c^3 = 1 + (s_2)^2/c^3$. It is easy to algebraically verify that for $c < 4^{1/3}$ we have $s_2 < s_3$.

Assuming Claim 2 is true for $k-1$, we derive

$$s_k = 1 + \sum_{i=1}^{k-1} x_i/x_k = 1 + \sum_{i=1}^{k-1} x_i \cdot \left(\sum_{i=1}^{k-1} x_i\right) / \left(c^3 \cdot (x_{k-1})^2\right) = 1 + \frac{(\sum_{i=1}^{k-1} x_i)^2}{c^3 (x_{k-1})^2} = 1 + (s_{k-1})^2/c^3.$$

Now suppose that $s_k \leq s_{k-1}$. Then by the induction hypothesis, we have $s_{k-1} - 1 - (s_{k-1})^2/c^3 \geq 0$. But when $c < 4^{1/3}$, differential calculus tells us that $x - 1 - x^2/c^3 < 0$, for all $x > 0$. Hence $s_k > s_{k-1}$, and the proof of Claim 2 is complete.

Finally, the increasing sequence $\{s_k\}$ is either unbounded, or it has a limit point. The first case cannot hold, as we have that $c^2 \geq s_k$ for all k, and c is a fixed constant. In the latter case, $\{s_k\}$ converges to the limit point $s_\infty = 1 + s_\infty^2/c^3$. The polynomial $p(x) = 1 + x^2/c^3 - x$ has roots $x = c^3/2 \cdot (1 \pm \sqrt{1 - 4/c^3})$. When $c < 4^{1/3}$, these roots are imaginary, therefore s_∞ would be imaginary, a contradiction. This concludes the proof. $\square$

An exhaustive computer search over proof annotations leads us to conjecture:

CONJECTURE 6.8. *There is no alternation trading proof that* $\mathsf{coNTIME}[n] \not\subseteq \mathsf{NTS}[n^c]$*, for any* $c > \sqrt[3]{4}$.

While this conjecture is still open, some interesting limitation can be proved. Namely, unlike the case of time lower bounds for SAT, we can show that no golden ratio lower bound can be achieved via alternation-trading proofs.

THEOREM 6.9. *There is no alternation trading proof that* $\mathsf{coNTIME}[n] \not\subseteq \mathsf{NTS}[n^c]$*, for any* $c \geq \phi \approx 1.618$.

PROOF. (Sketch) Suppose there is a proof that $\mathsf{coNTIME}[n] \not\subseteq \mathsf{NTS}[n^c]$ with $c \geq \phi$, and let A be a normal form annotation for it, of minimum length. First, observe that every valid annotation contains the sequence $1,0,0$ in it, for if every occurrence of $1,0$ was followed by a 1, the proof could not possibly be in normal form. (In particular, when a speedup rule and slowdown rule are applied to a simple class A, the resulting class A' has *more* quantifiers than A, in this setting.) Therefore $1,0,0$ must occur somewhere in the proof.

Next, we show that any subsequence $1,0,0$ can be removed from A, and the resulting LP will still be feasible for the constant c. This implies a contradiction.

Consider the four lines in a prospective proof corresponding to the sequence $1,0,0$, where we include the line before the three rules are applied. The first line is one of four possibilities:

$$\cdots^{b'} (\exists\, n^{a'})^b \mathsf{NTS}[n^a], \quad \cdots^{b'} (\forall\, n^{a'})^b \mathsf{coNTS}[n^a], \quad \cdots^{b'} (\exists\, n^{a'})^b \mathsf{coNTS}[n^a], \text{ or } \cdots^{b'} (\forall\, n^{a'})^b \mathsf{NTS}[n^a].$$

The first two cases are symmetric to each other, as are the last two cases, so it suffices for us to consider $\cdots^{b'} (\exists\, n^{a'})^b \mathsf{NTS}[n^a]$ and $\cdots^{b'} (\exists\, n^{a'})^b \mathsf{coNTS}[n^a]$.

In the first case, the four lines have the form

$$\cdots^{b'} (\exists\, n^{a'})^{b}\mathsf{NTS}[n^{a}] \tag{13}$$
$$\cdots^{b'} (\exists\, n^{\max\{a',x\}})^{\max\{b,x\}} (\forall\, \log n)^{b}\mathsf{NTS}[n^{a-x}] \tag{14}$$
$$\cdots^{b'} (\exists\, n^{\max\{a',x\}})^{\max\{b,x\}} (\forall\, \log n)^{b}\mathsf{coNTS}[n^{\max\{c(a-x),cb\}}] \tag{15}$$
$$\cdots^{b'} (\exists\, n^{\max\{a',x\}})^{\max\{b,x\}}\mathsf{NTS}[n^{\max\{c^2(a-x),c^2b,cx\}}] \tag{16}$$

Observe that each parameter in class (16) is at least the corresponding parameter in class (13), except for possibly the runtime of the NTS computation. However, if any one of $a \le c^2(a-x)$, $a \le c^2 b$, or $a \le cx$ hold, then the above lines can be removed from the proof, and the optimal assignment to the parameters would only be larger. So suppose $a > c^2(a-x)$, $a > c^2 b$, and $a > cx$. Then $c^2 a < a + c^2 x < a + ca$, implying that $c^2 < (1+c)$, or $c(c-1) < 1$. For $c \ge \phi$, this is a contradiction.

One can argue similarly for the second case. There the four lines have the form:

$$\cdots^{b'} (\exists\, n^{a'})^{b}\mathsf{coNTS}[n^{a}] \tag{17}$$
$$\cdots^{b'} (\exists\, n^{a'})^{b}(\forall\, n^{x})^{\max\{x,b\}}(\exists\, \log n)^{b}\mathsf{coNTS}[n^{a-x}] \tag{18}$$
$$\cdots^{b'} (\exists\, n^{a'})^{b}(\forall\, n^{x})^{\max\{x,b\}}(\exists\, \log n)^{b}\mathsf{NTS}[n^{\max\{c(a-x),cb\}}] \tag{19}$$
$$(\exists\, n^{a'})^{b}(\forall\, n^{x})^{\max\{x,b\}}\mathsf{coNTS}[n^{\max\{c^2(a-x),c^2b,cx\}}] \tag{20}$$

Using an argument similar to the above, class (20) contains class (17) when $c \ge \phi$, so removing the above lines can only improve the optimum setting of the parameters. □

7. LOWER BOUNDS FOR MULTIDIMENSIONAL TURING MACHINES

Our final case study considers lower bounds for a multidimensional machine model, which subsumes both the small-space random access model and off-line one-tape Turing machine models. The LP-based approach yields new lower bounds here as well. To recall, the machine model has a read-only/random-access input tape, a read-write/random access storage of $n^{o(1)}$ bits, and a d-dimensional tape that is read-write with sequential (two-way) access. This "hybrid" one-tape model holds importance in this area, since it is the most powerful deterministic model we know for which one can still prove an $n^{1+\Omega(1)}$ lower bound for solving satisfiability.

7.1. The Framework and Linear Programming Translation

Define $\mathsf{DTIME}_d[t(n)]$ to be the class of languages recognized by d-dimensional one-tape machines in $O(t(n))$ time. Lower bound proofs for these machines have different structure from the first two: one speedup rule simulates a d-dimensional machine by a nondeterministic (or co-nondeterministic) machine with a small space bound. More precisely, let Q represent a string of quantifier blocks, so $Q = (Q_1\ n^{a_1})^{b_2} \cdots (Q_k\ n^{a_{k-1}})$. The class $Q^b\mathsf{DTIME}_d[t(n)]$ consists of languages recognized by a machine which first takes k stages of guessing and writing $n^{a_i+o(1)}$ bits onto a write-once random-access storage (saving only n^{b_i} of these bits for the next stage). Once these k stages are completed, the machine then feeds $n^{b+o(1)}$ bits of content of the random-access storage (as well as the original input of length n) as input to a d-dimensional one-tape machine, which runs in time $t(n)$.

LEMMA 7.1 (DTIME_d TO DTISP). *Let $Q_{k+1} \in \{\exists,\forall\}$. Then for all $0 < s \le a$ and $b \ge 1$,*
$$Q^b\mathsf{DTIME}_d[n^a] \subseteq Q^b(Q_{k+1} n^{a-s})^{\max\{a-s,b\}}\mathsf{DTISP}[n^a, n^{ds}].$$

We can think of this as a "speedup lemma" where we have the added bonus of getting a DTISP class in the simulation of DTIME_d. The proof of Lemma 7.1 guesses a short crossing sequence that divides the $O(n^a)$ cells of tape used by a n^a-time machine into n^s blocks, each of which can be simulated in $O(n^{ds})$ space (see [Maass and Schorr 1987; van Melkebeek 2005]). We include a sketch of it for completeness.

PROOF. (Sketch) We prove the case where $d = 1$; the general case is similar. Let M be a machine where the final DTIME_d part runs in n^a time and x be an input of n^b length. For simplicity, we assume the two-way tape is infinite in one direction, and the tape has cells numbered $0, 1, 2, \ldots$.

We use a standard extension of the notion of crossing sequences. Define the *snapshot of $M(x)$ at timestep j* to be a tuple containing the state of M, the content of the $n^{o(1)}$ storage, the input head position, and the two-way tape head position. Note that a snapshot takes $n^{o(1)}$ bits to represent. The *snapshot sequence $\{s_k\}$ of $M(x)$* is the list of all snapshots of $M(x)$ in increasing order of timesteps. For a cell i of M's two-way tape, define the *crossing sequence for i on $M(x)$* to be the subsequence of the snapshot sequence of $M(x)$ containing only those s_k where either:

— the two-way tape head position is i in s_{k-1}, and the two-way tape head position is $i+1$ in s_k, or
— the two way-tape position is $i+1$ in s_{k-1}, and it is i in s_k.

That is, the crossing sequence for i represents all the information that is passed between cells i and $i+1$ during the computation.

Define $CS(x)$ to be the set of all crossing sequences for $M(x)$ on its sequential tape. Notice each element of a crossing sequence has $n^{o(1)}$ size.

For every $i = 1, \ldots, n^s$, define $CS(x,i)$ to be the subset of crossing sequences from $CS(x)$ ranging over the cells numbered $i + k \cdot n^s$, for $k = 0, 1, \ldots, n^{a-s} - 1$. That is, each $CS(x,i)$ contains a list of $O(n^{a-s})$ crossing sequences, where there are at least n^s cells between the sequences. Observe that the collection $\{CS(x,i)\}$ is a partition of $CS(x)$: the union of all $CS(x,i)$ equals $CS(x)$, and $CS(x,i) \cap CS(x,j) = \varnothing$ for $i \neq j$.

The total sum of the lengths of all crossing sequences for $M(x)$ is at most n^a (the total number of steps in the computation). Therefore there exists a j so that the total length of all crossing sequences in $CS(x,j)$ is at most n^{a-s}. If we guess $CS(x,j)$ upfront, then the computation of $M(x)$ reduces to an $n^{a+o(1)}$ time and $n^{s+o(1)}$ space computation, by checking that the computation between the crossing sequences of $CS(x,j)$ is a valid and accepting computation. Note that the guess can be existential or universal: if it is universal then we work with crossing sequences of the complement machine $\overline{M}$, and verify that $\overline{M}(x)$ does not accept, no matter which offset and n^{a-s} subset of crossing sequences that we try.

Let us sketch the case of arbitrary $d \geq 2$. Cells can be indexed by d-tuples of integers. Crossing sequences for a given cell $k = (i_1, \ldots, i_d)$ still make sense (keeping in mind that there are multiple cells in which the tape head can reach cell j). Analogously to the 1-dimensional case, we can define a partition of the set of all crossing sequences into n^{ds} parts, where each part breaks the tape into blocks of $O(n^{ds})$ cells – more precisely, they are $\underbrace{O(n^s) \times \cdots \times O(n^s)}_{d}$ cubes of cells. As before, the total sum of the lengths of all crossing sequences is n^a, and a counting argument shows that at least one part contains sequences with total length n^{a-s}. The space usage for simulation increases to $n^{ds+o(1)}$ because each block now has $O(n^{ds})$ cells. $\square$

Lemma 7.1 lets us apply the Speedup Lemma for space-bounded machines (Lemma 3.3) in our lower bound arguments for DTIME_d. We also need another Slowdown Lemma that lets us convert DTISP computations with nondeterministic bits back into DTIME_d computations. Again, let Q be a string of quantifier blocks.

LEMMA 7.2 (SLOWDOWN FOR DTIME_d). *Suppose* $\mathsf{NTIME}[n] \subseteq \mathsf{DTIME}_d[n^c]$. *Then for* $a_i, b_i \geq 1$*, and all* $e \geq 0$*,*

$$Q^{b_k}(Q_{k+1}\ n^{a_k})^{b_{k+1}}\mathsf{DTIME}_d[n^{a_{k+1}}] \subseteq Q^{b_k}\mathsf{DTIME}_d[n^{c \cdot \max\{b_k, b_{k+1}, a_k, a_{k+1}\}}], \text{ and}$$
$$Q^{b_k}(Q_{k+1}\ n^{a_k})^{b_{k+1}}\mathsf{DTISP}[n^{a_{k+1}}, n^e] \subseteq Q^{b_k}\mathsf{DTIME}_d[n^{c \cdot \max\{b_k, b_{k+1}, a_k, a_{k+1}\}}].$$

The proof directly follows that of earlier, similar results. We also use a time hierarchy theorem analogous to the previous lower bound settings:

LEMMA 7.3. *If* $\mathsf{NTIME}[n] \subseteq \mathsf{DTIME}_d[n^c]$ *for some c, then* $\mathsf{DTIME}_d[n^a] \nsubseteq \mathsf{DTIME}_d[n^{a'}]$ *for all* $a > a' \geq 1$.

PROOF. Analogous to Lemma 4.3, which showed that $\mathsf{NTIME}[n] \subseteq \mathsf{DTS}[n^c]$ implies $\mathsf{DTS}[n^a] \nsubseteq \mathsf{DTS}[n^{a'}]$, and Lemma 6.3, which showed that $\mathsf{NTIME}[n] \subseteq \mathsf{coNTS}[n^c]$ implies $\mathsf{NTS}[n^a] \nsubseteq \mathsf{coNTS}[n^{a'}]$. □

Example. In 1983, Kannan [Kannan 1983] proved that $\mathsf{NTIME}[n] \nsubseteq \mathsf{DTIME}_1[n\sqrt[4]{3/2}]$, using a weaker speedup lemma than the above. Reproducing his argument (but using the stronger speedup of Lemma 7.1 and Lemma 3.3):

$$\begin{aligned}
\mathsf{DTIME}_1[n^{3/2}] &\subseteq (\exists\, n)\mathsf{DTISP}[n^{3/2}, n^{1/2}] \quad \text{by } \mathsf{DTIME}_1 \text{ to } \mathsf{DTISP} \text{ (Lemma 7.1)} \\
&\subseteq (\exists\, n)(\forall\, \log n)\mathsf{DTISP}[n, n^{1/2}] \quad \text{by the Speedup Lemma for } \mathsf{DTISP} \text{ (Lemma 3.3)} \\
&\subseteq (\exists\, n)\mathsf{DTIME}_1[n^c] \quad \text{by the Slowdown Lemma for } \mathsf{DTIME}_d \text{ (Lemma 7.2)} \\
&\subseteq \mathsf{DTIME}_1[n^{c^2}] \quad \text{by Slowdown for } \mathsf{DTIME}_d
\end{aligned}$$

A contradiction follows from $c < \sqrt{3/2}$ and Lemma 7.3. But SAT $\in \mathsf{DTIME}_1[n^c]$ implies $\mathsf{NTIME}[n] \subseteq \mathsf{DTIME}_1[n^{c+o(1)}]$ (Theorem 3.1), so SAT cannot be solved in $O(n^{\sqrt{3/2}-\varepsilon})$ time on a one-dimensional Turing machine. This is the SAT lower bound proved by Van Melkebeek and Raz [van Melkebeek 2005].

Corresponding notions of alternation-trading proofs, proof annotations, and normal-form proofs can also be defined in this setting. The simple classes in this setting have the form

$$(Q_1\ n^{a_1})^{b_2} \ldots^{b_k} (Q_k\ n^{a_k})^{b_{k+1}} \mathscr{C}$$

where $\mathscr{C}$ can be a DTISP class or a DTIME_d class. When $\mathscr{C}$ is DTISP, the Speedup and Slowdown Rules from Section 4 apply; when $\mathscr{C}$ is a DTIME_d class, Lemma 7.1 and Lemma 7.2 are the possible applicable rules. Just as in the SAT time-space tradeoffs setting, there are two possible Speedup Rules for those classes with a DTISP phase: one rule introduces only a single quantifier, and the other rule introduces two (see Speedup Rules 1 and 2 of Section 4). However, analogous to Lemma 4.5, the Speedup Rule for DTISP which introduces two quantifiers can be shown to be unnecessary.

Therefore, for every simple class in an alternation-trading proof, there are only two possible rules to apply:

— **Speedup** via Lemma 3.3 when $\mathscr{C}$ is a DTISP class, or Lemma 7.1 (the DTIME_d to DTISP Lemma) when $\mathscr{C}$ is a DTIME_d class, or
— **Slowdown** via Lemma 3.4 when $\mathscr{C}$ is a DTISP class, or Lemma 7.2 when $\mathscr{C}$ is a DTIME_d class.

Hence the proof annotations for this setting can still be construed as bit vectors, for the two possible rules applicable at each step of a proof. (To illustrate, the annotation for the above example is $[1,1,0,0]$.) Notice that the structure of valid proof annotations becomes more complex: for example, when a class ends with DTISP and a Slowdown is applied, the new class has one less quantifier but now ends with DTIME_d instead.

A *normal form proof for c* in this setting has $\mathsf{DTIME}_d[n^a]$ as its first class and $\mathsf{DTIME}_d[n^{a'}]$ as its last class, where $a' \leq a$. Analogously to Lemma 4.3 (for DTS lower bounds) and Lemma 6.5 (for NTS lower bounds), and building on Lemma 7.3, normal form proofs can be shown to entail lower bounds:

THEOREM 7.4. *Let A_1 and A_2 be simple and complementary. If there is an alternation-trading proof P that* ($\mathsf{NTIME}[n] \subseteq \mathsf{DTIME}_d[n^c] \implies A_1 \subseteq A_2$)*, then there is a normal form proof for c, of length at most that of P.*

The proof closely follows that of Theorems 4.4 and 6.6. Finally, the translation of proof annotations to linear programming is also very similar, except that new variables s_i are introduced for the space exponent of the DTISP class in line i (for all relevant i). The linear inequalities relating the s_i variables to other variables are taken directly from the statements of Lemma 7.2 and Lemma 7.1.

7.2. Empirical Results

For 1-dimensional machines, a summary of lower bounds found by the LP-based theorem prover is given in the below table. Unlike the previous two cases, the optimal bounds attained by short proofs have non-monotonic behavior (with respect to length) at first. Perhaps surprisingly, the table looks the same for the 2-dimensional and 3-dimensional cases, albeit with smaller lower bound exponents.

#Lines	Best Proof Annotation(s)	L.B.
5	[1, 1, 0, 0]	1.224
6	[1, 1, 0, 1, 0]	
7	[1, 1, 1, 0, 0, 0]	1.201
8,9	[1, 1, 0, 1, 1, 0, 0], [1, 1, 0, 1, 1, 0, 1, 0]	1.262
10	[1, 1, 1, 0, 0, 1, 1, 0, 0]	1.261
11, 12	[1, 1, 0, 1, 1, 0, 1, 1, 0, 0], [1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0]	1.274
13	[1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0]	1.277
14, 15	[1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0],[1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0]	1.278
16, 17	[1, 1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0], [1, 1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 0]	1.287
19	[1, 1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0]	1.292
25	[1, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0]	1.297
28	[1, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0]	1.298

A subset of the optimal annotations have the form

$$A = [1\ (1\ 1\ 0)^k\ 0\ (1\ 1\ 0)^\ell\ 0],$$

for integers k, ℓ. In other words, the substring 0 0 occurs exactly twice, and in all optimal annotations found, 0 0 occurs at most twice. Might it be that for longer proofs, there are optimal annotations with three occurrences of 0 0? As before, we used a heuristic search to investigate. The search uncovered more interesting annotations, but all of the best had the form of A above. For instance, the best 25 line proof was:

```
0, DTIME1[n^1.751958454]
1, (E n^1.)DTISP[n^1.751958454,n^.7519608720]
2, (E n^1.040108911)(A n^1.)DTISP[n^1.463810415,n^.7519608720]
3, (E n^1.040108911)(A n^1.)(E n^1.)DTISP[n^1.215771287,n^.7519608720]
4, (E n^1.040108911)(A n^1.)DTIME1[n^1.577881470]
5, (E n^1.040108911)(A n^1.)DTISP[n^1.577881470,n^.5778814720]
6, (E n^1.040108911)(A n^1.)(E n^1.)DTISP[n^1.155762944,n^.5778814720]
7, (E n^1.040108911)(A n^1.)DTIME1[n^1.5]
8, (E n^1.040108911)(A n^1.)DTISP[n^1.5,n^.5]
9, (E n^1.040108911)(A n^1.)(E n^1.)DTISP[n^1.,n^.5]
10, (E n^1.040108911)(A n^1.)DTIME1[n^1.297844000]
11, (E n^1.040108911)DTIME1[n^1.684399048]
12, (E n^1.040108909)DTISP[n^1.684399048,n^.6442901394]
13, (E n^1.040108909)(A n^1.)DTISP[n^1.288580278,n^.6442901394]
14, (E n^1.040108909)DTIME1[n^1.672376183]
15, (E n^1.040108909)DTISP[n^1.672376183,n^.6322672739]
16, (E n^1.040108909)(A n^1.)DTISP[n^1.264534548,n^.6322672739]
```

```
17, (E n^1.040108909)DTIME1[n^1.641168576]
18, (E n^1.040108909)DTISP[n^1.641168576,n^.6010596669]
19, (E n^1.040108911)(A n^1.)DTISP[n^1.202119332,n^.6010596669]
20, (E n^1.040108911)DTIME1[n^1.560163362]
21, (E n^1.040108911)DTISP[n^1.560163362,n^.5200544533]
22, (E n^1.040108908)(A n^1.)DTISP[n^1.040108908,n^.5200544533]
23, (E n^1.040108908)DTIME1[n^1.349899105]
24, DTIME1[n^1.751958454]
```

The best annotation found (for all dimensions d) was the 66 line annotation

$$[1,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,0,\\ 1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,0],$$

which leads to proofs that $\mathsf{NTIME}[n]$ is not in any of $\mathsf{DTIME}_1[n^{1.3009}]$, $\mathsf{DTIME}_2[n^{1.1875}]$, and $\mathsf{DTIME}_3[n^{1.1343}]$. In fact, all the best annotations found had the form $[1\ 1\ 1\ (0\ 1\ 1)^k\ 0\ (0\ 1\ 1)^\ell\ 0\ 0]$.

7.3. Interpreting The Results

The annotations suggest a proof where one has an inductive lemma capturing the $(0\ 1\ 1)^*$ behavior, then applies the lemma twice to a proof of six more lines. This strategy leads to:

THEOREM 7.5. *SAT* $\notin \mathsf{DTIME}_d[n^c]$, *for all* $c < r_d$ *where* $r_d \geq 1$ *is a root of*

$$p_d(x) = (2d+1)(d+1)^2x^5 - 2(d+1)(2d+1)x^4 - d^2x^3 + 2(d+1)x^2 - ((2d+1)(d+1)^2+1)x + d(d+1).$$

As corollaries, we have SAT $\notin \mathsf{DTIME}_1[n^{1.3009}]$, SAT $\notin \mathsf{DTIME}_2[n^{1.1887}]$, and SAT $\notin \mathsf{DTIME}_3[n^{1.1372}]$. Before we prove Theorem 7.5, we first give an inductive lemma. Let $c \geq 1$, and define the sequence

$$e_1 := \frac{d+2}{d+1},\ \ e_{k+1} := 1 + \frac{e_k}{c(d+1)}.$$

PROPOSITION 7.6. *If* $c < e_1 = (d+2)/(d+1)$ *then* $\{e_k\}$ *is increasing.*

PROOF. By induction on k. If $c < e_1 = (d+2)/(d+1) = 1 + 1/(d+1)$, then $e_2 = 1 + (e_1/c)/(d+1) > e_1$, and $e_{k+1} = 1 + (e_k/c)/(d+1) > 1 + (e_{k-1}/c)/(d+1) = e_k$. □

LEMMA 7.7. *Let* $c < (d+2)/(d+1)$. *If* $\mathsf{NTIME}[n] \subseteq \mathsf{DTIME}_d[n^c]$, *then for all k,*

$$\mathsf{DTIME}_d[n^{e_k}] \subseteq (\exists\ n)(\forall\ \log n)\mathsf{DTISP}[n, n^{d/(d+1)}].$$

PROOF. When $k = 1$,

$$\mathsf{DTIME}_d[n^{e_k}] \subseteq (\exists\ n)\mathsf{DTISP}[n^{(d+2)/(d+1)}, n^{d/(d+1)}] \subseteq (\exists\ n)(\forall\ \log n)\mathsf{DTISP}[n, n^{d/(d+1)}],$$

by Lemma 7.1 (DTIME_d to DTISP) and Lemma 3.3 (DTISP Speedup), respectively.

For the inductive step,

$$\begin{aligned}\mathsf{DTIME}_d[n^{1+\frac{e_k}{c(d+1)}}] &\subseteq (\exists\ n)\mathsf{DTISP}[n^{1+\frac{e_k}{c(d+1)}}, n^{\frac{de_k}{c(d+1)}}] \quad (\mathsf{DTIME}_d \text{ to } \mathsf{DTISP})\\ &\subseteq (\exists\ n)(\forall\ \log n)\mathsf{DTISP}[n^{e_k/c}, n^{\frac{de_k}{c(d+1)}}] \quad \text{(Speedup)}\\ &\subseteq (\exists\ n)\mathsf{DTIME}_d[n^{e_k}] \quad (\mathsf{DTIME}_d \text{ Slowdown, and } e_k > c \text{ by Proposition 7.6})\\ &\subseteq (\exists\ n)(\exists\ n)(\forall\ \log n)\mathsf{DTISP}[n, n^{d/(d+1)}] = (\exists\ n)(\forall\ \log n)\mathsf{DTISP}[n, n^{d/(d+1)}],\end{aligned}$$

where the last containment holds by induction. □

Note the proof annotations for the derivations in the above lemma have the form $(1\ 1\ 0)^{k-1}\ 1\ 1$, as suggested by experiments.

COROLLARY 7.8. *Let $c < (d+2)/(d+1)$. If $\mathsf{NTIME}[n] \subseteq \mathsf{DTIME}_d[n^c]$ then for all $\varepsilon > 0$,*

$$\mathsf{DTIME}_d[n^{1+\frac{1}{c(d+1)-1}-\varepsilon}] \subseteq (\exists\, n)(\forall\, \log n)\mathsf{DTISP}[n, n^{d/(d+1)}].$$

PROOF. For $e < 1 + \frac{1}{c(d+1)-1}$, we have $e < 1 + \frac{e}{c(d+1)-1}$. The sequence $e_k = 1 + \frac{e_{k-1}}{c(d+1)}$, $e_1 = (d+2)/(d+1)$ converges to $\eta = 1 + \frac{1}{c(d+1)-1}$ for all $c \geq 1$. (Note that $e = 1 + e/(c(d+1))$ implies $e = 1 + \frac{1}{c(d+1)-1}$.)

Therefore, for any $e^* < \eta$, by setting $e = (d+1)/(d+2)$ and observing

$$\mathsf{DTIME}_d[n^{(d+1)/(d+2)}] \subseteq (\exists\, n)(\forall\, \log n)\mathsf{DTISP}[n, n^{d/(d+1)}]$$

(the base case of Lemma 7.7), one can apply Lemma 7.7 for a constant number of times, and infer that the same containment holds for $\mathsf{DTIME}_d[n^{e^*}]$. □

Intuitively, the corollary says that as we make stronger assumptions about how quickly SAT can be solved on a d-dimensional one-tape TM, then we can simulate more of the class $\mathsf{DTIME}_d[n^{O(1)}]$ in the class $(\exists\, n)(\forall\, \log n)\mathsf{DTISP}[n, n^{d/(d+1)}]$, when $c < (d+2)/(d+1)$. We can now prove the lower bound.

Proof of Theorem 7.5. Let $a \geq 1$ be a parameter and let $c < (d+2)/(d+1)$. Assuming SAT is in $\mathsf{DTIME}_d[n^c]$, we will find a smaller setting of c which yields a contradiction. Then

$$\begin{aligned}\mathsf{DTIME}_d[n^a] &\subseteq (\exists\, n)\mathsf{DTISP}[n^a, n^{d(a-1)}] \quad (\mathsf{DTIME}_d \text{ to } \mathsf{DTISP})\\ &\subseteq (\exists\, n^{x+d(a-1)})^{x+d(a-1)}(\forall\, \log n)^1\mathsf{DTISP}[n^{a-x}, n^{d(a-1)}] \quad \text{(Speedup)}\end{aligned}$$

where x is a parameter satisfying $c \geq x + d(a-1) \geq 1$. By Speedup, the above class is in

$$\begin{aligned}&(\exists\, n^{x+d(a-1)})^{x+d(a-1)}(\forall\, n^{(1-d(a-1))+d(a-1)})^1(\exists\, \log n)^1\mathsf{DTISP}[n^{a-x-(1-d(a-1))}, n^{d(a-1)}]\\ =\ &(\exists\, n^{x+d(a-1)})^{x+d(a-1)}(\forall\, n^1)^1(\exists\, \log n)^1\mathsf{DTISP}[n^{a-x-(1-d(a-1))}, n^{d(a-1)}]\\ \subseteq\ &(\exists\, n^{x+d(a-1)})^{x+d(a-1)}(\forall\, n^1)^1\mathsf{DTIME}_d[n^{c(a-x-(1-d(a-1)))}] \quad \text{(Slowdown)},\end{aligned}$$

assuming (for the moment) that $1 - d(a-1) \geq 0$. Suppose that a and x satisfy $c(a - x - (1 - d(a-1))) = c((d+1)(a-1) - x) \geq 1 + \frac{1}{c(d+1)-1} - \varepsilon$, for some $\varepsilon > 0$. Applying Corollary 7.8, the above is contained in

$$\begin{aligned}&(\exists\, n^{x+d(a-1)})(\forall\, n)(\exists\, \log n)\mathsf{DTISP}[n^1, n^{d/(d+1)}]\\ \subseteq\ &(\exists\, n^{x+d(a-1)})(\forall\, n)\mathsf{DTISP}[n^c, n^{d/(d+1)}] \quad \text{(Slowdown)}\\ \subseteq\ &(\exists\, n^{x+d(a-1)})^{x+d(a-1)}\mathsf{DTIME}_d[n^{c^2}], \quad \text{(Slowdown)}\end{aligned}$$

since $c \geq x + d(a-1)$. Now suppose a and c satisfy $c^2/(x + d(a-1)) = 1 + 1/(c(d+1)-1) - \varepsilon$. Then Corollary 7.8 can be applied again, obtaining the class

$$\begin{aligned}&(\exists\, n^{x+d(a-1)})(\forall\, n)\mathsf{DTISP}[n^{(x+d(a-1))}, n^{(x+d(a-1))\cdot\frac{d}{d+1}}]\\ \subseteq\ &(\exists\, n^{x+d(a-1)})\mathsf{DTIME}_d[n^{c(x+d(a-1))}] \quad \text{(Slowdown)}\\ \subseteq\ &\mathsf{DTIME}_d[n^{c^2(x+d(a-1))}]. \quad \text{(Slowdown)}.\end{aligned}$$

Setting $a = c^2(x + d(a-1))$ yields a contradiction with Lemma 7.4. Observe that a proof annotation for the above has the form $[1\ (1\ 1\ 0)^k\ 0\ (1\ 1\ 0)^\ell\ 0]$. The analysis introduced three parameters (c, a,

x) along with three equations to satisfy:

$$a = c^2(x+d(a-1)), \tag{21}$$

$$c((d+1)(a-1)-x) = 1+\frac{1}{c(d+1)-1}, \tag{22}$$

$$\frac{c^2}{x+d(a-1)} = 1+\frac{1}{c(d+1)-1}. \tag{23}$$

For the sake of completeness, let us sketch how this system is solved and how the polynomial $p_d(x)$ arises. Equation (21) is a linear equation in a, so we can easily derive

$$a = \frac{c^2(d-x)}{c^2d-1}.$$

Substituting into equations (22) and (23), we have

$$\frac{c+cx+cd-2c^3dx-c^3x}{c^2d-1} = 1+\frac{1}{c(d+1)-1} \tag{24}$$

and

$$\frac{c^2(c^2d-1)}{d-x} = 1+\frac{1}{c(d+1)-1}. \tag{25}$$

Noting that x is linear in equation (24), we obtain

$$x = \frac{c(d+1)(d+1-cd)}{(cd+c-1)(2c^2d+c^2-1)}.$$

Substituting into equation 25, we have

$$\frac{(2c^2d+c^2-1)c^2(cd+c-1)}{(2cd^2+3cd+c-d)} = 1+\frac{1}{c(d+1)-1}.$$

Multiplying denominators on both sides, then subtracting the RHS from both sides, we obtain

$$\frac{c\cdot p_d(c)}{(2cd^2+3cd+c-d)(cd+c-1)} = 0,$$

where

$$p_d(x) = (2d+1)(d+1)^2x^5-2(d+1)(2d+1)x^4-d^2x^3+2(d+1)x^2-((2d+1)(d+1)^2+1)x+d(d+1).$$

For $c>1$ and $d\geq 1$, the denominator is nonzero and positive. Therefore, any $c\geq 1$ which is a root of $p_d(x)$ will suffice. For any $r<c$, we can find a, x, and $\varepsilon>0$ satisfying

$$r(d(a-1)-x) = 1+1/(r(d+1)-1)-\varepsilon,$$

$$a \geq r^2(x+d(a-1)),$$

$$r^2/(x+d(a-1)) = 1+1/(r(d+1)-1)-\varepsilon.$$

This completes the proof. □

We again conjecture that the above lower bound is optimal for alternation-trading proofs. We can show that no $n^{1+1/(d+1)}$ lower bound can be proved for d-dimensional TMs under the current system of rules.

THEOREM 7.9. *There is no alternation trading proof that SAT* $\notin \mathsf{DTIME}_d[n^c]$, *for any* $c\geq 1+1/(d+1)$.

PROOF. (Sketch) The proof is by minimal counterexample. Suppose there is an alternation-trading proof that $\mathsf{NTIME}[n] \not\subseteq \mathsf{DTIME}_d[n^c]$ with $c \geq 1+1/(d+1)$, and let A be a normal form annotation. Let us first give a complete proof that A cannot be $[1,1,0,0]$, by showing that this annotation cannot yield $c \geq 1+1/(d+1)$. The annotation $[1,1,0,0]$ refers to a proof with the following structure (where $x,y \geq 0$):

$$\begin{aligned} &\mathsf{DTIME}_d[n^a] \\ &(\exists\, n^{a-x})^{\max\{a-x,1\}}\mathsf{DTISP}[n^a,n^{dx}] \\ &(\exists\, n^{\max\{a-x,y+dx\}})^{\max\{a-x,y+dx,1\}}(\forall\, n^0)^1\mathsf{DTISP}[n^{a-y},n^{dx}] \\ &(\exists\, n^{\max\{a-x,y+dx\}})^{\max\{a-x,y+dx,1\}}\mathsf{DTIME}_d[n^{c\cdot\max\{a-x,y+dx,1,a-y\}}] \\ &\mathsf{DTIME}_d[n^{c^2\cdot\max\{a-x,y+dx,1,a-y\}}] \end{aligned}$$

Let $a' = c^2 \cdot \max\{a-x, y+dx, 1, a-y\}$. For convenience, let $c' = c^2$. In order for $a \geq a'$ (to yield a lower bound, by Theorem 7.4), we must have

$$a \geq c', \tag{26}$$

$$a \geq c'(dx+y), \tag{27}$$

$$a \geq c'(a-x), \tag{28}$$

$$a \geq c'(a-y). \tag{29}$$

(28) and (29) imply

$$a \leq c'x/(c'-1) \quad a \leq c'y/(c'-1). \tag{30}$$

Combining with (27), we have $dx+y \leq x/(c'-1)$ and $dx+y \leq y/(c'-1)$, hence $y \leq x(1/(c'-1)-d)$ and $x \leq y(1/(c'-1)-1)/d$. That is, $y/x \leq (1/(c'-1)-d)$ and $y/x \geq d/(1/(c'-1)-1)$, therefore

$$\frac{1}{c'-1} - d \geq \frac{d}{1/(c'-1)-1}.$$

This condition directly implies that $c' \leq (d+2)/(d+1)$, or $c \leq \sqrt{1+1/(d+1)} < 1+1/(d+1)$.

From here on, we may assume that $|A| > 4$, otherwise the only valid annotation is $[1,1,0,0]$ which we know does not yield a strong lower bound.

Now we prove that every sequence $0,1,0$ in a normal form annotation can be replaced with just $0,0$. After a slowdown, the deterministic portion of a class is DTIME_d, therefore $0,1,0$ produces the lines:

$$\begin{aligned} &\cdots^{b'}(\exists\, n^{a'})^b\mathsf{DTIME}_d[n^a] \\ &\cdots^{b'}(\exists\, n^{\max\{a',a-s\}})^{\max\{b,a-s\}}\mathsf{DTISP}[n^a,n^{ds}] \\ &\cdots^{b'}\mathsf{DTIME}_d[n^{\max\{cb,ca-cs,ca',ca,cds\}}]. \end{aligned}$$

But these lines give no improvement over applying the sequence $0,0$ instead, since that would replace the above with

$$\begin{aligned} &\cdots^{b'}(\exists\, n^{a'})^b\mathsf{DTIME}_d[n^a] \\ &\cdots^{b'}\mathsf{DTIME}_d[n^{\max\{cb,ca,ca'\}}], \end{aligned}$$

but $\max\{cb, ca-cs, ca', ca, cds\} \geq \max\{cb, ca, ca'\}$.

Once we have removed all occurrences of $0,1,0$, the proof annotations can be placed in 1-1 correspondence with strings of balanced parentheses of the form (x), as in Theorem 4.10. Informally, each (corresponds to the addition of a new quantifier by some speedup rule, and each) corresponds to the removal of a quantifier, by a slowdown rule. More formally, after the first run of k 1's in an annotation (which produces a line with k quantifier blocks, and hence k open parentheses in the string), every subsequent run of k 1's corresponds to only $k-1$ open parentheses (because the first 1 in a run corresponds to the switch from DTIME_d to DTISP, which does not add a new quantifier if the class already has quantifiers), and every 0 corresponds to a closed parenthesis, removing one quantifier.

Now we make the following claim:

CLAIM 3. *After removing all occurrences of* $0,1,0$ *in an annotation, there must be always be an occurrence of the subsequence* $1,1,0,0$ *in every valid annotation with at least four lines.*

To prove the claim, we first show that the annotation must end with the sequence $0,0$. The last bit of the annotation must have a 0 (so that a DTIME_d class is reached). Assume (for a contradiction) that the bit before that is 1 (a speedup). Then the corresponding class has a DTISP predicate and only one quantifier, so the next-to-last class has the form

$$(\exists\, n^{a'})^b \mathsf{DTISP}[n^a, n^e].$$

But then the bit before this must have been 0; it could not be 1, because then we must have applied the DTISP speedup rule and introduced at least two quantifiers. Hence the last three bits of the annotation must be $0,1,0$, which cannot be. Therefore the annotation must end with $0,0$.

Now we argue for the existence of $1,1,0,0$. There must be at least two 1's in the first run of 1's in the annotation, otherwise the annotation cannot be in normal form: it would reach the class $\mathsf{DTIME}_d[n^a]$ before the last line. Hence there is a substring $1,1,0$ indicating the end of the first run of 1's. There are three cases:

— If the next bit is a 0, then $1,1,0,0$ has been found.
— If the next two bits are $1,0$, then there is a subsequence $0,1,0$, which cannot be.
— If the next two bits are $1,1$, this signals the beginning of a new run of at least two 1's; we continue the argument with the sequence $1,1,0$ at the end of this run of 1's.

Because the end of the annotation has the sequence $0,0$, we must eventually reach a substring $1,1,0,0$. This concludes the proof of the claim.

Finally, we claim that if $c \geq (d+2)/(d+1)$, then any sequence $1,1,0,0$ in an annotation can be replaced by either the sequence $1,0$, or the sequence 0. There are two cases to consider.

Case 1. The line before the $1,1,0,0$ sequence has the form:

$$Q^b(Q_1\, n^{a'})^b \mathsf{DTISP}[n^a, n^e].$$

Then, the lines corresponding to $1,1,0,0$ have the form:

$$\begin{aligned}
&Q^b(Q_1\, n^{a'})^b \mathsf{DTISP}[n^a, n^e]\\
&Q^b(Q_1\, n^{\max\{a',x+e\}})^{\max\{b,x+e\}}(Q_2\, n^0)^{\max\{b,1\}} \mathsf{DTISP}[n^{a-x}, n^e]\\
&Q^b(Q_1\, n^{\max\{a',x+e\}})^{\max\{b,x+e\}}(Q_2\, n^y)^{\max\{b,y+e,1\}}(Q_3\, n^0)^{\max\{b,1\}} \mathsf{DTISP}[n^{a-x-y}, n^e]\\
&Q^b(Q_1\, n^{\max\{a',x+e\}})^{\max\{b,x+e\}}(Q_2\, n^y)^{\max\{b,y+e,1\}} \mathsf{DTIME}_d[n^{c\cdot\max\{b,y+e,1,a-x-y\}}]\\
&Q^b(Q_1\, n^{\max\{a',x+e\}})^{\max\{b,x+e\}} \mathsf{DTIME}_d[n^{c^2\cdot\max\{b,y+e,1,x+e,a-x-y\}}]
\end{aligned}$$

In comparison, the string $1,0$ would have the form:

$$\begin{aligned}
&Q^b(Q_1\, n^{a'})^b \mathsf{DTISP}[n^a, n^e]\\
&Q^b(Q_1\, n^{\max\{a',x+e\}})^{\max\{b,x+e\}}(Q_2\, n^0)^{\max\{b,1\}} \mathsf{DTISP}[n^{a-x}, n^e]\\
&Q^b(Q_1\, n^{\max\{a',x+e\}})^{\max\{b,x+e\}} \mathsf{DTIME}_d[n^{c\cdot\max\{b,1,x+e,a-x\}}]
\end{aligned}$$

Similar to previous proofs (Theorem 4.10 and 6.9), we infer that when

$$\max\{b,1,x+e,a-x\} \leq c\cdot\max\{b,y+e,1,x+e,a-x-y\}, \tag{31}$$

we can replace $1,1,0,0$ with $1,0$. If one of b, $x+e$, or 1 is the maximum on the LHS, then for any $c \geq 1$, the inequality (31) certainly holds. Now suppose $a-x$ is the maximum on the LHS. Letting $y=0$, we see that $c \cdot \max\{b, y+e, 1, x+e, a-x-y\} \geq c(a-x) \geq a-x$ for every $c \geq 1$. Hence, no matter what a', a, b, e are, (31) holds for all $c \geq 1$.

Case 2. The line before the $1,1,0,0$ sequence has the form:

$$Q^b(Q_1\ n^{a'})^b \mathsf{DTIME}_d[n^a].$$

Notice that, if the line has this form, then the previous rule must have been a slowdown, because every application of a speedup rule results in a line with DTISP. This case is similar to the base case above, where we considered the annotation $[1,1,0,0]$ directly. The lines have the form:

$$\begin{aligned} Q^b(Q_1\ n^{a'})^b \mathsf{DTIME}_d[n^a] \\ Q^b(Q_1\ n^{\max\{a',a-x\}})^{\max\{b,a-x\}} \mathsf{DTISP}[n^a, n^{dx}] \\ Q^b(Q_1\ n^{\max\{a',a-x,y+dx\}})^{\max\{a-x,y+dx,b\}} (\forall\ n^0)^1 \mathsf{DTISP}[n^{a-y}, n^{dx}] \\ Q^b(Q_1\ n^{\max\{a',a-x,y+dx\}})^{\max\{a-x,y+dx,b\}} \mathsf{DTIME}_d[n^{c \cdot \max\{a-x,y+dx,b,a-y\}}] \\ Q^b \mathsf{DTIME}_d[n^{c^2 \cdot \max\{a',a-x,y+dx,b,a-y\}}]. \end{aligned}$$

We want to show that $c \geq (d+2)/(d+1)$ implies

$$c^2 \cdot \max\{a', a-x, y+dx, b, a-y\} \geq c \cdot \max\{a, a', b\}.$$

Then, we can always replace $1,1,0,0$ by 0 without affecting the proof. Notice that if a' and b are the maximum on the RHS, then the inequality is trivially true, so suppose that a is the maximum. Dividing both sides by c, the following inequalities are implied:

$$a \leq c, \tag{32}$$

$$a \leq c(dx+y), \tag{33}$$

$$a \leq c(a-x), \tag{34}$$

$$a \leq c(a-y). \tag{35}$$

This is extremely similar to the system we solved for the annotation $[1,1,0,0]$, except for two points: (a) the directions of the inequalities are reversed, and (b) in the previous system, we had the variable $c' = c^2$. By analogous reasoning, $c \geq (d+2)/(d+1)$ yields a feasible solution to the system, hence $1,1,0,0$ can be replaced by 0 in this case.

Therefore, if $c \geq (d+2)/(d+1)$ is proved by an annotation, then that annotation can always be made shorter, without affecting the lower bound proof. Hence every annotation can be reduced to one of at most four lines, which we know does not suffice for the lower bound. □

8. CONCLUSION

We introduced a methodology for reasoning about lower bounds in the alternation-trading framework. This gives a general way to attack lower bound problems via computer, and lets us establish limitations on known techniques. We now have a better understanding of what these techniques can and cannot do, and a tool for addressing future problems. Previously, the problem of setting parameters to get a good lower bound was a highly technical exercise. Our work should reduce the load on further research: once a new speedup or slowdown lemma is found, one only needs to find the relevant linear programming formulation to begin understanding its power. We end with two open problems.

(1) *Establish tight limitations for alternation-trading proofs.* That is, show that the best possible alternation-trading proofs match the ones we have provided. The computer search results of this paper have been met with skepticism; it is critical to verify these perceived limitations with formal proof. As mentioned earlier, this open problem has been resolved in the case of DTISP lower bounds on SAT [Buss and Williams 2012].

(2) *Discover ingredients that add signficantly to the framework.* Here there are several possible avenues. One is to find new separation results that lead to new contradictions. Another is to find improved Speedup and/or Slowdown Lemmas. The Slowdown Lemmas appear to be "blandest" of the ingredients: they are very elementary, and they relativize. For instance, there may be a better Speedup Lemma that shows DTS is contained *infinitely often* in a faster alternating time class, and we might use an *almost-everywhere* time hierarchy [Geske et al. 1991; Allender et al. 1993] to obtain a contradiction.
Secondly, combinatorial methods have led to several time-space lower bounds. For example, Ajtai [Ajtai 2002] and Beame *et al.* [Beame et al. 2001] have proven time-space lower bounds for branching programs; Gurevich and Shelah [Gurevich and Shelah 1988] gave a problem in $\mathsf{NTISP}[n, \log n]$ but not $\mathsf{DTISP}[n^{1+a}, n^b]$) when $b + 2a < 1/2$. Is it possible to incorporate these combinatorial methods into the alternation-trading framework?

9. ACKNOWLEDGEMENTS

I am grateful to my thesis committee for their invaluable feedback on my PhD thesis, which included some preliminary results on this work. Thanks also to Scott Aaronson for useful discussions about "irrelativization" and to Sam Buss for helpful comments. I am grateful to anonymous referees for their feedback on earlier versions of this work. The TOCT referee was especially thorough, and their comments substantially improved the presentation.

REFERENCES

E. Allender, R. Beigel, U. Hertrampf, and S. Homer. Almost-Everywhere Complexity Hierarchies for Nondeterministic Time. *Theor. Comput. Sci.* 115(2):225–241, 1993.

E. Allender, M. Koucky, D. Ronneburger, S. Roy, and V. Vinay. Time-Space Tradeoffs in the Counting Hierarchy. In *Proceedings of IEEE Conference on Computational Complexity (CCC)*, 295–302, 2001.

D. B. Arnold and M. R. Sleep. Uniform random generation of balanced parenthesis strings. *ACM Transactions on Programming Languages and Systems* 2(1):122–128, 1980.

S. Arora and B. Barak. *Computational Complexity: A Modern Approach.* Cambridge University Press, 2009.

M. Ajtai. Determinism versus Nondeterminism for Linear Time RAMs with Memory Restrictions. *J. Computer and System Sciences* 65:2–37, 2002.

P. Beame, T. S. Jayram, and M. E. Saks. Time-Space Tradeoffs for Branching Programs. *J. Computer and System Sciences* 63(4):542–572, 2001.

S. Buss and R. Williams. Limits on Alternation-Trading Proofs for Time-Space Lower Bounds. In *Proceedings of IEEE Conference on Computational Complexity (CCC)*, 181–191, 2012.

A. K. Chandra, D. Kozen, and L. J. Stockmeyer. Alternation. *JACM* 28(1):114–133, 1981.

S. A. Cook. Short Propositional Formulas Represent Nondeterministic Computations. *Information Processing Letters* 26(5): 269-270, 1988.

C. Derman. *Finite State Markov Decision Processes.* Academic Press, 1972.

S. Diehl and D. van Melkebeek. Time-Space Lower Bounds for the Polynomial-Time Hierarchy on Randomized Machines. *SIAM J. Computing* 36:563–594, 2006.

S. Diehl, D. van Melkebeek, and R. Williams. An Improved Time-Space Lower Bound for Tautologies. *J. Comb. Optim.* 22(3):325–338, 2011.

M. Dietzfelbinger and M. Hühne. Matching upper and lower bounds for simulations of several linear tapes on one multidimensional tape. *Computational Complexity* 8(4): 371–392, 1999.

L. Fortnow. Nondeterministic Polynomial Time Versus Nondeterministic Logarithmic Space: Time-Space Tradeoffs for Satisfiability. In *Proceedings of IEEE Conference on Computational Complexity (CCC)*, 52–60, 1997.

L. Fortnow and D. van Melkebeek. Time-Space Tradeoffs for Nondeterministic Computation. In *Proceedings of IEEE Conference on Computational Complexity (CCC)*, 2–13, 2000.

L. Fortnow, R. Lipton, D. van Melkebeek, and A. Viglas. Time-Space Lower Bounds for Satisfiability. *JACM* 52(6):835–865, 2005.

J. G. Geske, D. T. Huynh, and J. I. Seiferas. A Note on Almost-Everywhere-Complex Sets and Separating Deterministic-Time-Complexity Classes. *Inf. Comput.* 92(1):97–104, 1991.

D. Yu. Grigor'ev. Time complexity of multidimensional Turing machines. *J. Mathematical Sciences* 20(4):2290–2295, 1982.

Y. Gurevich and S. Shelah. Nondeterministic Linear-Time Tasks May Require Substantially Nonlinear Deterministic Time in the Case of Sublinear Work Space. *JACM* 37(3):674–687, 1990.

J. Y. Halpern, M. C. Loui, A. R. Meyer, and D. Weise. On Time versus Space III. *Mathematical Systems Theory* 19(1):13–28, 1986.

J. Hopcroft, W. Paul, and L. Valiant. On time versus space. *JACM* 24(2):332–337, 1977.

R. Kannan. Alternation and the power of nondeterminism. In *Proceedings of ACM Symposium on Theory of Computing (STOC)*, 344–346, 1983.

R. Kannan. Towards Separating Nondeterminism from Determinism. *Mathematical Systems Theory* 17(1):29–45, 1984.

R. J. Lipton and A. Viglas. On the Complexity of SAT. In *Proceedings of IEEE Symposium on Foundations of Computer Science (FOCS)*, 459–464, 1999.

M. Liskiewicz and K. Lorys. Fast Simulations of Time-Bounded One-Tape Turing Machines by Space-Bounded Ones. *SIAM J. Computing* 19(3):511–521, 1990.

M. C. Loui. Simulations among multidimensional Turing machines. Ph.D. Thesis, Massachusetts Institute of Technology TR-242, 1980.

W. Maass and A. Schorr. Speed-Up of Turing Machines with One Work Tape and a Two-Way Input Tape. *SIAM J. Computing* 16(1):195–202, 1987.

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.

D. van Melkebeek. A Survey of Lower Bounds for Satisfiability and Related Problems. *Foundations and Trends in Theoretical Computer Science* 2(3):197–303, 2006.

D. van Melkebeek and R. Raz. A Time Lower Bound for Satisfiability. *Theoretical Computer Science* 348(2-3):311–320, 2005.

D. van Melkebeek and T. Watson. A quantum time-space lower bound for the counting hierarchy. Technical Report 1600, Department of Computer Sciences, University of Wisconsin-Madison, 2007.

V. Nepomnjascii. Rudimentary predicates and Turing calculations. *Soviet Math. Doklady* 11:1462–1465, 1970.

W. Paul and R. Reischuk. On time versus space II. *J. Computer and System Sciences* 22:312–327, 1981.

W. Paul, N. Pippenger, E. Szemeredi, and W. Trotter. On determinism versus nondeterminism and related problems. In *Proceedings of IEEE Symposium on Foundations of Computer Science (FOCS)*, 429–438, 1983.

R. Santhanam. Lower bounds on the complexity of recognizing SAT by Turing machines. *Information Processing Letters* 79(5):243–247, 2001.

C. Schnorr. Satisfiability is quasilinear complete in NQL. *JACM* 25(1):136–145, 1978.

I. Tourlakis. Time-Space Tradeoffs for SAT on Nonuniform Machines. *J. Computer and System Sciences* 63(2):268–287, 2001.

E. Viola. On approximate majority and probabilistic time. In *Proceedings of the 22th IEEE Conference on Computational Complexity (CCC)*, 2007.

R. Williams. Inductive Time-Space Lower Bounds for SAT and Related Problems. *J. Computational Complexity* 15:433–470, 2006.

R. Williams. Algorithms and Resource Requirements for Fundamental Problems. Ph.D. Thesis, Carnegie Mellon University, CMU-CS-07-147, August 2007.

R. Williams. Time-Space Tradeoffs for Counting NP Solutions Modulo Integers. *J. Computational Complexity* 17(2), 2008.