Journal of Computer and System Sciences **63**, 268–287 (2001)
doi:10.1006/jcss.2001.1767, available online at http://www.idealibrary.com on IDEAL®

Time–Space Tradeoffs for SAT on Nonuniform Machines[1]

Department of Computer Science, Princeton University, Princeton, New Jersey
E-mail: itourlak@cs.princeton.edu

Received August 31, 2000; revised February 2, 2001

The arguments used by R. Kannan (1984, *Math. Systems Theory* **17**, 29–45), L. Fortnow (1997, *in* "Proceedings, Twelfth Annual IEEE Conference on Computational Complexity, Ulm, Germany, 24–27 June, 1997," pp. 52–60), and R. J. Lipton and A. Viglas (1999, *in* "40th Annual Symposium on Foundations of Computer Science, New York, 17–19 Oct. 1999," pp. 459–469) are generalized and combined with an argument for diagonalizing over machines taking n bits of advice on inputs of length n to obtain the first nontrivial time–space lower bounds for **SAT** on nonuniform machines. In particular, we show that for any $a<\sqrt{2}$ and any $\varepsilon>0$, **SAT** cannot be computed by a random access deterministic Turing machine using n^a time, $n^{o(1)}$ space, and $o(n^{\sqrt{2}/2-\varepsilon})$ advice nor by a random access deterministic Turing machine using $n^{1+o(1)}$ time, $n^{1-\varepsilon}$ space, and $n^{1-\varepsilon}$ advice. More generally, we show that if for some $\varepsilon>0$ there exists a random access deterministic Turing machine solving **SAT** using n^a time, n^b space, and $o(n^{(a+b)/2-\varepsilon})$ advice, then $a\geqslant\frac{1}{2}(\sqrt{b^2+8}-b)$. Lower bounds for computing $\overline{\mathbf{SAT}}$ on random access nondeterministic Turing machines taking sublinear advice are also obtained. Moreover, we show that **SAT** does not have NC^1 circuits of size $n^{l+o(1)}$ generated by a nondeterministic log–space machine taking $n^{o(1)}$ advice. Additionally, new separations of uniform classes are obtained. We show that for all $\varepsilon>0$ and all rational numbers $r\geqslant 1$, DTISP$(n^r, n^{1-\varepsilon})$ is properly contained in NTIME(n^r). © 2001 Academic Press

Key Words: computational complexity; time-space trade-offs; nonuniform complexity, circuit complexity.

[1] Supported in part by NSERC Fellowship PGSA- 208465-1998. Work done while at the Department of Computer Science at the University of Toronto, Toronto, Canada.

1. INTRODUCTION

A central problem in complexity theory is to determine the relationship between P and NP. In [8] Cook showed that to separate these classes, it suffices to show that a deterministic machine requires superpolynomial time to recognize the set **SAT** of satisfiable propositional formulas. Consequently, obtaining lower bounds for **SAT** is one of the main goals of complexity theory. While it is expected that **SAT** requires exponential deterministic time and exponential size circuits, even superlinear lower bounds have proved elusive.

An important first step was made in [11] where Fortnow showed that **SAT** cannot be computed by a deterministic machine running in time $n^{1+o(1)}$ if the machine is also restricted to $n^{1-\varepsilon}$ space for any $\varepsilon > 0$. More generally, Fortnow showed that for all $\varepsilon > 0$ and any unbounded function $r(n) \in O(\frac{\log n}{\log \log n})$,

$$\overline{\textbf{SAT}} \notin \text{NTIME}\,(n^{1+\frac{1}{r(n)}}) \cap \text{NTISP}(n^{o(r(n))}, n^{1-\varepsilon}).$$

In addition, Fortnow's paper presented a proof (credited to Harry Buhrman) showing that **SAT** cannot be computed by log–space uniform NC^1 circuits of size $n^{1+o(1)}$.

Later, Lipton and Viglas [21], using techniques similar to the ones used by Fortnow, showed that for all $\delta > 0$, **SAT** cannot be computed by a deterministic Turing machine running in time $n^{\sqrt{2}-\delta}$ and polylogarithmic space. In addition, [21] claimed that **SAT** $\notin$ DTISP$(n^{2-\delta}, \log^{O(1)} n)$ for all $\delta > 0$; however, this result was subsequently retracted [28].

Both the Fortnow and Lipton–Viglas results built on techniques used by Kannan [18] to show the existence of an integer k such that for all integers $j \neq k$, NTIME(n^j) properly contains DTISP$(n^j, o(n^{j/k}))$. Indeed, all these results combine diagonalization arguments with work showing that if **SAT** or, more generally, nondeterministic computations, have fast deterministic algorithms, then alternations in an alternating computation can be removed with only a small loss in efficiency.

By reversing the order of the alternations in the Lipton–Viglas arguments, Fortnow and van Melkebeek [12] were able to obtain further improvements. For instance, they were able to show that for any constant a less than the golden ratio, **SAT** cannot be solved in n^a time and $n^{o(1)}$ space.

1.1. Statement of Main Results

In this paper the emphasis is on obtaining time–space tradeoffs for **SAT** on machines taking sublinear advice. The arguments used by Kannan, Fortnow, and Lipton and Viglas are generalized and combined with a general argument for diagonalizing over machines taking n bits of advice on inputs of length n to obtain partially nonuniform time–space lower bounds for **SAT**.

The diagonalization argument is presented in Section 3. In Section 4, partially nonuniform extensions to the Fortnow–Buhrman bounds in [11] are obtained with the aid of this diagonalization argument and by adapting the proofs in [11] to the nonuniform setting. Recall that a function f is *subpolynomial* if $f(n) \in n^{o(1)}$. Note that $\log^k n$ and $\log^{\log\log n} n$ are subpolynomial. In Section 4.1, Buhrman's lower bound is generalized to show that **SAT** does not have NC^1 circuits of size $n^{1+o(1)}$ generated by a nondeterministic log–space transducer taking subpolynomial advice. In Section 4.2, Fortnow's lower bound is generalized to show that for all unbounded functions $r(n) \in O(\frac{\log n}{\log\log n})$ and any $\varepsilon > 0$,

$$\overline{\mathbf{SAT}} \notin \mathrm{NTIME}(n^{1+\frac{1}{r(n)}})/n^{o(1)} \cap \mathrm{NTISP}(n^{o(r(n))}, n^{1-\varepsilon})/n^{o(1)}.$$

In Section 5 the proof framework in Lipton and Viglas [21] is adapted and combined with the diagonal result from Section 3 to provide even stronger time–space lower bounds for linear nondeterministic computations on deterministic machines taking sublinear advice. In particular, it is shown that for all ε, $0 < \varepsilon \leqslant 1$, if

$$\mathrm{NTIME}(n) \subseteq \mathrm{DTISP}(n^{\alpha}, n^{\varepsilon})/o(n^{\frac{1}{2}(\alpha+\varepsilon)}), \tag{1}$$

then $\alpha \geqslant \frac{1}{2}(\sqrt{\varepsilon^2+8}-\varepsilon)$. Equation (1) can then be used to obtain lower bounds for **SAT**. In particular, for all $\delta > 0$,

$$\mathbf{SAT} \notin \mathrm{DTISP}(n^{1+o(1)}, n^{1-\delta})/n^{1-\delta},$$
$$\mathbf{SAT} \notin \mathrm{DTISP}(n^{1.186}, \sqrt{n})/n^{0.843},$$
$$\mathbf{SAT} \notin \mathrm{DTISP}(n^{\sqrt{2}-\delta}, n^{o(1)})/o\left(n^{\frac{\sqrt{2}}{2}-\delta}\right).$$

More generally, Eq. (1) implies that for all $\delta > 0$ and any ε, $0 < \varepsilon < 1$, **SAT** cannot be computed by a random access deterministic Turing machine using $n^{1/2(\sqrt{\varepsilon^2+8}-\varepsilon)-\delta}$ time, n^{ε} space and $o(n^{1/4(\sqrt{\varepsilon^2+8}+\varepsilon-\delta)})$ advice.

Equation (1) is a special case of a general result proved in Section 5 which gives time–space tradeoffs for NTIME(n^r) for all rational $r \geqslant 1$ on deterministic machines taking sublinear advice. This result shows that for all rational numbers α, r, and ε such that $\alpha \geqslant 1$, $r \geqslant 1$, $0 < \varepsilon \leqslant 1/r$, and $\alpha + \varepsilon < \frac{r+1}{r}$, if

$$\mathrm{NTIME}(n^r) \subseteq \mathrm{DTIME}(n^{\alpha r}, n^{\varepsilon r})/o(n^{(\alpha+\varepsilon)/(r+1)}), \tag{2}$$

then $\alpha \geqslant \frac{1}{2}(\sqrt{\varepsilon^2+4+4/r}-\varepsilon)$. Another application of this more general result, in addition to the one above, is to show that for all rational $r \geqslant 1$ and all $\varepsilon > 0$,

$$\mathrm{DTISP}(n^r, n^{1-\varepsilon}) \subsetneqq \mathrm{NTIME}(n^r).$$

The latter result is similar to Kannan's result [18] mentioned above. However, as the constant in Kannan's result is unknown, the exact relationship with Kannan's result is uncertain.

Time–space tradeoffs for co-nondeterministic time on nondeterministic machines taking sublinear advice are obtained in Section 6. The following lower bounds for $\overline{\mathrm{SAT}}$ are subsequently derived:

$$\overline{\mathbf{SAT}} \notin \mathrm{NTISP}(n^{1.142}, \sqrt{n})/O(n^{0.499}),$$
$$\overline{\mathbf{SAT}} \notin \mathrm{NTISP}(n^{1.324}, n^{o(1)})/n^{o(1)}.$$

1.2. Related Work

There exists a large body of work on time–space tradeoffs. The first such result was probably Cobham's time–space tradeoff for the recognition of the set of palindromes [7]. Dúriś and Galil [10], Karchmer [19], and Gurevich and Shelah [14] obtained time–space tradeoffs by taking advantage of the inefficiencies of computation models that read their input sequentially. Beame [3], Beame *et al.* [5], and Ajtai [1] have obtained nonuniform time–space tradeoffs on branching programs. With a remarkable technical breakthrough, Ajtai [2] subsequently proved a superlinear lower bound on the depth of any subexponential size deterministic branching programs computing a function based on quadratic forms. Building on this work, Beame *et al.* [4] obtained the first time–space tradeoffs for randomized computation.

There are several separations in addition to the ones mentioned above related to the time–space tradeoffs discussed in this paper. In many cases, the proofs of these separations are the foundations of many of the time–space tradeoffs mentioned above and presented in this paper. One such paper is [16] where Hopcroft *et al.* showed that for sufficiently "nice" $t(n)$, DTIME(t) is properly contained in DSPACE(t). Building on the techniques in [16], Paul *et al.* [24] showed that DTIME(n) is properly contained in NTIME(n) for (nonrandom access) multitape Turing machines. Building on the techniques in [16] and [24], Gupta [13] then subsequently showed that for time-constructible functions $t(n)$, DTIME($t(n) \log^* t(n)$) is properly contained in $\Sigma_2^{t(n)}$. As a corollary, it follows that either NTIME(n) $\neq$ coNTIME(n) or DTIME($t(n)$) is properly contained in NTIME($t(n)$) for all time-constructible $t(n)$. Unlike the related results in this paper, the results in the above three papers only hold for standard multitape Turing machines rather than for random access machines. Nevertheless, all three of these papers share some of the basic ideas used in this paper as well as in the series of results inspired by Kannan's and Fortnow's papers [18, 11].

2. PRELIMINARIES

The model of computation used is the random access multitape Turing machine as defined in [12]. Such machines are like normal multitape Turing machines except that they have two types of tapes: index and nonindex tapes. With the exception of the output tape, the tapes come in pairs. Each nonindex tape T (other than the output tape) has an associated index tape I_T such that the machine can move its head on T to the position indicated by I_T in one step. The heads on the index tapes can only be moved one unit to the left or right in one step. One of the random access nonindex tapes is read-only and contains the input for the machine. If the machine has output, it is equipped with a unidirectional write-only output tape. The length of an index tape is restricted to $O(\log t)$ where t is the running time of the machine. The space used by a machine is the largest nonindex tape location indexed. The benefits from using this model of computation come from the fact that these machines seem to be able to simulate more realistic machines (like RAMs) with only polylogarithmic losses in efficiency while still being easy to work with. Hence, results on these machines hold for other more general models modulo polylogarithmic factors. In the remainder of the paper, "Turing machine" and "machine" will refer to random access Turing machines unless explicitly stated otherwise.

Familiarity with the concept of alternation as well as with the definitions and basic properties of standard complexity classes is assumed. Background material can be found in [17, 23].

Let DTISP$(t(n), s(n))$ denote the class of all languages accepted by deterministic Turing machines running simultaneously in $O(t(n))$ time and $O(s(n))$ space. Let NTISP$(t(n), s(n))$ be the nondeterministic version.

Only time-constructible functions will be considered in this paper. A function f, $f(n) \in \Omega(\log n)$, is *time-constructible* if there exists an $O(f(n))$ time-bounded deterministic Turing machine that on inputs of length n outputs $f(n)$ in binary. In this paper, $\log n$ denotes the integer-valued function $\lceil \log_2 n \rceil$ whose value at 0 is defined to be 0.

Define $\langle x, y \rangle$ to be the string formed by doubling each bit in x and appending $01y$ to the resulting string. Note that $|\langle x, y \rangle| = 2\,|x| + |y| + 2$. Note also that $\langle , \rangle$ and its inverse can both be computed extremely efficiently.

The nonuniform classes considered in this paper are defined using "advice functions" as in Karp and Lipton [20]:

Definition 2.1 [20]. Let $\mathscr{C}$ be a class of languages and let $\mathscr{F}$ be a collection of functions mapping $\mathbb{N}$ to $\mathbb{N}$. A language L belongs to the nonuniform class $\mathscr{C}/\mathscr{F}$ if there exist a language L' in $\mathscr{C}$ and a function $h\colon \mathbb{N} \to \Sigma^*$ such that $|h| \in \mathscr{F}$ and $x \in L$ if and only if $\langle x, h(|x|) \rangle \in L'$.

The functions $\mathscr{F}$ in the above definition are called *advice functions*. In the following the set $\mathscr{F}$ in a class $\mathscr{C}/\mathscr{F}$ is restricted to time-constructible functions. Moreover, it is assumed that if $f \in \mathscr{F}$, then all time-constructible functions g such that $g \leqslant f$ pointwise are also in $\mathscr{F}$. By abusing notation, $\mathscr{C}/a(n)$ will be used to denote the class $\mathscr{C}/\mathscr{F}$ where $\mathscr{F}$ consists of all time-constructible functions g such that $g < a$ pointwise.

Let NC^1[NL$/n^{o(1)}$] consist of all languages L for which there exist a family $\{C_n\}$ of polynomial size logarithmic depth circuits accepting L, a nondeterministic log–space Turing machine M with output, and a function $h\colon \mathbb{N} \to \Sigma^*$ such that $|h| \in n^{o(1)}$ and M on input $\langle 1^n, h(n) \rangle$ outputs C_n. Note that NC^1[NL$/n^{o(1)}$] properly contains log–space uniform NC^1.

A $\Sigma_{a(n)}^{t(n)}$ machine is an alternating random access Turing machine that runs in time $t(n)$ and makes $a(n) - 1$ alternations beginning with $\exists$. Let $\Sigma_{a(n)}^{t(n)}$ be the class of languages accepted by $\Sigma^{t(n)}_{a(n)}$ machines. Recall that a quantified Boolean formula is a formula with an initial block of quantifiers followed by a quantifier-free formula. Let $\mathbf{QBF}_{a(n)}$ be the set of true quantified formulas having $a(n)-1$ quantifier alternations beginning with $\exists$. Recall that $\mathbf{QBF}_{a(n)}$ is complete for $\Sigma^p_{a(n)}$. The *length* of a quantified Boolean formula is the number of bits needed to express the formula in a fixed standard paddable encoding. The *size* of a quantified Boolean formula is the number of variables it contains. Boolean formulas will sometimes be referred to as propositional formulas.

Let M be a Turing machine. Denote by $L(M)$ the language accepted by M. Denote by $\bar{M}$ the machine that behaves just like M except that if M enters an accepting (rejecting) state, then $\bar{M}$ enters a rejecting (accepting) state, and if M enters an existential (universal) state, then $\bar{M}$ enters a universal (existential) state. Note that $L(\bar{M}) = \overline{L(M)}$. A configuration A of M describes the machine's state along with the contents of its work-tapes together with the positions of its input- and work-tape heads. Let $A \vdash_M B$ be true if M can reach configuration B from configuration A in one step. More generally, $A \vdash^k_M B$ is true if M starting in A can reach B in k steps.

2.1. *Reducing Computations to SAT*

In [9] Cook showed that a nondeterministic multitape (nonrandom access) Turing machine computation of time $t(n)$ can be reduced to a **SAT** question of size $O(t(n) \log t(n))$. Cook's result built on work by Hennie and Stearns [15] and Pippenger and Fischer [25] showing that a deterministic (nonrandom access) multitape Turing machine computation of time $t(n)$ can be simulated by a family $\{C_n\}$ of circuits with $O(t(n) \log t(n))$ gates. Cook's result can be extended to alternating machines making a constant number of alternations:

LEMMA 2.1 (Cook). *Let M be a $t(n)$ time-bounded alternating multitape Turing machine making a constant number a of alternations on input of length n. Then there exists a function ϕ mapping inputs x, $|x| = n$, to quantified propositional formulas of length $O(t(n) \log^2 t(n))$ such that M accepts x iff $\phi(x)$ is in $\mathbf{QBF}_a$. Moreover, ϕ is computable in $O(t(n) \log^2 t(n))$ time and $O(\log t(n))$ space, and a machine that has random access to x can compute the ith bit of $O(x)$ in $O(\log^{O(1)} t(n))$ time and $O(\log t(n))$ space.*

In [27] Robson showed that nondeterministic RAM computations of time $t(n)$, where the time complexity is measured using the logarithmic cost criterion, can be reduced to **SAT** questions of size $O(t(n) \log t(n))$. Robson's proof can be adapted to show that for each $t(n)$ time-bounded nondeterministic *random access* Turing machine M there exists a function ϕ mapping inputs x, $|x| = n$, to propositional formulas of size $O(t(n) \log t(n))$ such that M accepts x iff $\phi(x)$ is in **SAT**. For the results in this paper it is necessary for the formulas $\phi(x)$ to be easy to compute in the sense that a machine with random access to x can compute the ith bit of $\phi(x)$ using only polylogarithmic time and logarithmic space.

Now, the formulas in Robson's reduction consist of parts representing the computation of M together with a component for asserting that a given list L_2 is the stable sort of another list L_1. Whereas it is easy to see that the components of Robson's formulas not corresponding with the stable-sort are easy to compute in the sense identified above, it is not clear that the parts of the formulas representing the stable sort are sufficiently easy to compute. The stable sort components of a formula in Robson's reduction have size $O(t(n) \log t(n))$. This component can be replaced by a new formula of size $O(t(n) \log^2 t(n))$ that is sufficiently easy to compute in the sense required above as follows. Consider a (nonrandom access) multitape Turing machine M' that accepts iff its input consists of two lists where the second list is a merge-sort of the first list. Such a machine can be made to run in $O(n \log n)$ time. Lemma 2.1 then implies the existence of a function ϕ' mapping inputs y, $|y| = n$, to propositional formulas of size $O(n \log n)$ such that $\phi'(y)$ is in **SAT** iff y consists of two lists where the second list is a merge-sort of the first list. Moreover, if one has random access to y, the ith bit of $\phi'(y)$ can be computed using polylogarithmic time and logarithmic space. Noting that merge-sort is a stable sort, the following lemma can be proved by replacing in Robson's reduction the part representing a stable sort with the formula obtained via ϕ':

LEMMA 2.2. *Let M be a $t(n)$ time-bounded nondeterministic random access Turing machine. Then there exists a function ϕ mapping inputs x, $|x| = n$, to propositional formulas of length $O(t(n) \log^3 t(n))$ such that M accepts x iff $\phi(x)$ is in* **SAT**. *Moreover, a machine that has random access to x can compute the ith bit of $\phi(x)$ in $O(\log^{O(1)} t(n))$ time and $O(\log t(n))$ space.*

More generally, and by appropriately padding formula encodings, the following lemma can be proved:

LEMMA 2.3. *Let M be a $\Sigma_a^{t(n)}$ machine. Then there exist a constant D and a function ϕ mapping inputs x, $|x| = n$, to quantified propositional formulas of length $Dt(n) \log^3 t(n)$ such that M accepts x iff $\phi(x)$ is in* $\mathbf{QBF}_a$. *Moreover, a machine that has random access to x can compute the ith bit of $\phi(x)$ in $O(\log^{O(1)} t(n))$ time and $O(\log t(n))$ space.*

Lemma 2.3 is used in this paper to obtain lower bounds for **SAT**. Note that Lemma 2.3 holds for random access Turing machines whereas Lemma 2.1 holds only for standard nonrandom access Turing machines. The stronger Lemma 2.3 is required to obtain lower bounds for random access machines.

3. DIAGONALIZATION

The new technical tool used in this paper is the following theorem together with variations of it. The proof refines an argument used by Rackoff [26] to construct a language in DTIME($(n \log n)$ $(s(n) \log^2 s(n))$) that cannot be computed by circuits of size $s(n)$ having easily decodable descriptions of size n.

THEOREM 3.1. *Let $t(n)$ and $T(n)$ be functions such that $t(3n+2) \in o(T(n))$. Then*

$$\mathrm{NTIME}(T(n)) \nsubseteq \mathrm{coNTIME}(t(n))/n.$$

Proof. Fix an enumeration of all conondeterministic Turing machines where all machines appear infinitely often and consider a nondeterministic Turing machine U that computes as follows: on input x, $|x| = n$, U first computes $T(n)$. Then U simulates the complement $\overline{M_n}$ of the nth machine M_n on input $\langle x, x\rangle$ for $T(n)$ steps. Computation branches of U that do not have enough time to finish the simulation of $\overline{M_n}$ reject after $T(n)$ steps. It is clear that $L = L(U)$ is in NTIME$(T(n))$.

The claim is that L is not in coNTIME$(t(n))/n$. Indeed, suppose to the contrary that there exists a co-nondeterministic Turing machine M running in time $t(n)$ and a function h mapping $\mathbb{N}$ to $\{0, 1\}^*$ such that $|h(n)| = n$ and x is in L iff M accepts $\langle x, h(|x|)\rangle$. First note that in the same way that one shows that a two-tape nondeterministic Turing machine can simulate a multitape Turing machine with only a constant factor slowdown [6], it can be shown that a random access two-tape nondeterministic Turing machine can simulate a random access multitape Turing machine with only a constant factor slowdown. Hence, U can simulate n steps of $\bar{M}$'s computation in cn steps where c only depends on M. Let m be a sufficiently large index such that $M_m = M$ and $ct(3m+2) \leqslant T(m)$. Consider U on input $h(m)$. By definition, U simulates $\overline{M_m} = \bar{M}$ on input $\langle h(m), h(m)\rangle$. The simulation of $\bar{M}$ on $\langle h(m), h(m)\rangle$ requires time $ct(3m+2) \leqslant T(m)$, and thus, U has enough time to complete the simulation. But then, $h(m)$ is in L iff U accepts $h(m)$ iff $\bar{M}$ accepts $\langle h(m), h(m)\rangle$ iff M rejects $\langle h(m), h(m)\rangle$ iff $h(m)$ is not in L, a contradiction. ∎

Note that the term $3n+2$ in the theorem is simply an artifact of the definition of the pairing function $\langle , \rangle$. Note also that the arguments in the proof of Theorem 3.1 are quite general and can be used to show analogous separations between other complexity classes. For instance, the arguments can be used to show that $\Sigma_{s(n)}^{n \log n}$ is not contained in $\Sigma_{O(1)}^{n}/n$ when $s(n)$ is unbounded, a result that will be used in the proof of Theorem 4.1 below.

4. EXTENDING THE FORTNOW–BUHRMAN LOWER BOUNDS

4.1. Circuit Lower Bounds for SAT

The main theorem proved in this subsection is an extension of Buhrman's lower bound for **SAT**:

THEOREM 4.1. **SAT** *cannot both have circuits of size* $n^{1+o(1)}$ *and be in* $\mathrm{NL}/n^{o(1)}$. *In particular,* **SAT** *cannot be computed by* $\mathrm{NC}^1[\mathrm{NL}/n^{o(1)}]$ *circuits of size* $n^{1+o(1)}$.

The proof of Theorem 4.1 follows the proof given in [11] for Buhrman's original result. Like in Buhrman's original result, the proof relies on the following two lemmas. The first appeared in Fortnow [11] where it was credited to Buhrman. The second follows easily from the version of Nepomnjaščiĭ's theorem [22] presented in Kannan [18] and later in Fortnow [11].

LEMMA 4.1 (Buhrman). *Suppose that* **SAT** *has circuits of size* $n^{1+o(1)}$. *Then* $\mathbf{QBF}_{s(n)} \in \Sigma_2^p$ *for some unbounded function* $s(n) \in O(\frac{\log n}{\log \log n})$.

LEMMA 4.2. (Nepomnjaščiĭ's theorem) *Let $\varepsilon > 0$. Then for any functions $t(n)$ and $\alpha(n)$, $1 \leqslant \alpha(n) \leqslant n^{o(1)}$,*

$$\mathrm{NTISP}(n^{O(\alpha(n))}, n^{1-\varepsilon})/t(n) \subseteq \Sigma^{n}_{O(\alpha(n))}/t(n).$$

Proof (Proof of Theorem 4.1). Assume **SAT** has circuits of size $n^{1+o(1)}$. It then follows from Lemma 4.1 that

$$\Sigma^{n \log n}_{s(n)} \subseteq \Sigma^{p}_{2}. \tag{3}$$

Assume in addition that $\mathbf{SAT} \in \mathbf{NL}/n^{o(1)}$ to derive a contradiction. The first step is to show that this assumption implies that $\Sigma^{p}_{2} \subseteq \mathrm{NL}/n^{o(1)}$. To that end, note that any language in $\mathrm{NP}/n^{o(1)}$ can be reduced to **SAT** via a log–space transducer taking subpolynomial advice. So if $\mathbf{SAT} \in \mathrm{NL}/n^{o(1)}$, it follows that $\mathrm{NP}/n^{o(1)} = \mathrm{NL}/n^{o(1)}$. In particular, $\mathrm{NP}/n^{o(1)} = \mathrm{coNP}/n^{o(1)}$.

Now fix $A \in \Sigma^{p}_{2}$. There exists a polynomial f and a relation $R \in \mathrm{coNP}$ such that $x \in A$ iff there exists y, $|y| < f(|x|)$, such that $(x, y) \in R$ (assume that $|(x, y)| = |x| + |y|$). Moreover, by padding y, assume without loss of generality that $|y| = f(|x|)$. Since $\mathrm{coNP}/n^{o(1)} = \mathrm{NP}/n^{o(1)}$, it follows that $R \in \mathrm{NP}/n^{o(1)}$. So there exist a polynomial time nondeterministic machine M and a function $h\colon \mathbb{N} \to \Sigma^{*}$ such that $|h|$ is subpolynomial and $(x, y) \in R$ iff M accepts $\langle (x, y), h(|(x, y)|)\rangle$. Consider a nondeterministic machine M' that on input $\langle x, w\rangle$, checks that $|w| = |h(|x| + f(|x|))|$, guesses a string y of length $f(|x|)$, and then simulates M on $\langle (x, y), w\rangle$. Then M' runs in polynomial time and $x \in A$ iff M' accepts $\langle x, h(|x| + f(|x|))\rangle$. Moreover, since $|h|$ is subpolynomial, $|h(n + f(n))|$ is also subpolynomial and thus, $A \in \mathrm{NP}/n^{o(1)} = \mathrm{NL}/n^{o(1)}$. So $\Sigma^{p}_{2} \subseteq \mathrm{NL}/n^{o(1)}$.

Equation (3) and Lemma 4.2 then imply that

$$\Sigma^{n \log n}_{s(n)} \subseteq \Sigma^{p}_{2} \subseteq \mathrm{NL}/n^{o(1)} \subseteq \Sigma^{n}_{O(1)}/n^{o(1)}.$$

A contradiction is now obtained by finding a language in $\Sigma^{n \log n}_{s(n)} - (\Sigma^{n}_{O(1)}/n)$ by generalizing the arguments in the proof of Theorem 3.1. ▮

Note that the proof of Theorem 4.1 can be adapted to show that **SAT** cannot both have circuits of size $n^{1+o(1)}$ and be in $\mathrm{DTISP}(n^{O(1)}, n^{o(1)})/n^{o(1)}$. From this it follows that **SAT** does not have circuits of quasilinear size and subpolynomial width where the circuits are generated by a deterministic machine using polynomial time and subpolynomial space and advice.

4.2. Extending Fortnow's Lower Bound

By utilizing a diagonal argument similar to the one used in the proof of Theorem 3.1, the following extension to Fortnow's time–space tradeoff for $\overline{\mathbf{SAT}}$ can be proved:

THEOREM 4.2. *Let $r(n)$ be any unbounded function such that $r(n) \in O(\frac{\log n}{\log\log n})$. Then for all $\varepsilon > 0$,*

$$\overline{\mathrm{SAT}} \notin \mathrm{NTIME}(n^{1+\frac{1}{r(n)}})/n^{o(1)} \cap \mathrm{NTISP}(n^{o(r(n))}, n^{1-\varepsilon})/n^{o(1)}.$$

In particular, for any $\varepsilon > 0$,

$$\overline{\mathrm{SAT}} \notin \mathrm{NTISP}(n^{1+o(1)}, n^{1-\varepsilon})/n^{o(1)}.$$

The arguments used to prove Theorem 4.2 follow along the same lines as those used by Fortnow to prove his original result. The differences lie in the introduction of the different diagonal argument to extend to the nonuniform case.

For his result Fortnow required the following lemma which gives sufficient conditions for collapsing a nonconstant number of levels of the polynomial hierarchy:

LEMMA 4.3 (Fortnow). *Let $r(n) \in O(\frac{\log n}{\log\log n})$ be unbounded and suppose that $\overline{\mathrm{SAT}} \in \mathrm{NTIME}(n^{1+1/r(n)})$. Then for any $\gamma > 0$, there exists a constant ℓ such that*

$$\Sigma^{n\log n}_{\frac{r(n)}{\ell}} \subseteq \mathrm{coNTIME}(n^{1+\gamma}).$$

To prove Theorem 4.2 the following nonuniform version of Fortnow's lemma is required:

LEMMA 4.4. *Let $r(n) \in O(\frac{\log n}{\log\log n})$ be an unbounded function and suppose that $\overline{\mathrm{SAT}} \in \mathrm{NTIME}(n^{1+1/r(n)})/n^{o(1)}$. Then for any $\gamma > 0$, there exists a constant ℓ such that*

$$\Sigma^{n\log n}_{\frac{r(n)}{\ell}} \subseteq \mathrm{coNTIME}(n^{1+\gamma})/n^{o(1)}.$$

It is straightforward to verify that Fortnow's proof for Lemma 4.3 can be adapted to prove the lemma.

Proof (Proof of Theorem 4.2). Fix $r(n)$ and ε, and let ℓ be the constant guaranteed by Lemma 4.4 for $\gamma = \varepsilon/4$. Let $\alpha(n) \in o(r(n))$ and construct a language L in

$$\Sigma^{n\log n}_{\frac{r(n)}{\ell}} - (\Sigma^{n}_{O(\alpha(n))}/n^{o(1)})$$

by arguing as in the proof of Theorem 3.1. Now assume that

$$\overline{\mathrm{SAT}} \in \mathrm{NTIME}(n^{1+\frac{1}{r(n)}})/n^{o(1)} \cap \mathrm{NTISP}(n^{o(r(n))}, n^{1-\varepsilon})/n^{o(1)}.$$

A contradiction will be derived by showing that L is in $\Sigma^{n}_{O(\alpha(n))}/n^{o(1)}$.

Since $\overline{\mathrm{SAT}} \in \mathrm{NTIME}(n^{1+1/r(n)})/n^{o(1)}$, it follows from Lemma 4.4 that $\bar{L}$ is in $\mathrm{NTIME}(n^{1+\gamma})/n^{o(1)}$. In particular, there exist a nondeterministic Turing machine

M_1 running in time $n^{1+\gamma}$ and a function h: $\mathbb{N} \to \Sigma^*$ such that $|h(n)| \in n^{o(1)}$ and $x \in L$ iff M_1 accepts $\langle x, h(|x|)\rangle$. By Lemma 2.3, there exists a constant b such that for each input x, $|x| = n$, there exists a formula $\phi(x)$ of length $bn^{1+2\gamma}$ such that x is in $L(M_1)$ iff $\phi(x)$ is satisfiable, and moreover, the ith bit of $\phi(x)$ can be computed in polylogarithmic time and logarithmic space provided random access to x.

Since $\overline{\mathbf{SAT}} \in \mathrm{NTISP}(n^{\alpha(n)}, n^{1-\varepsilon})/n^{o(1)}$, there exist a nondeterministic machine M_2 running in $n^{\alpha(n)}$ time and using $n^{1-\varepsilon}$ space and a function h': $\mathbb{N} \to \Sigma^*$ such that $|h'(n)| \in n^{o(1)}$ and $x \in \overline{\mathbf{SAT}}$ iff M_2 accepts $\langle x, h(|x|)\rangle$.

Now consider a nondeterministic machine M_3 that on input $\langle x, h'(b\,|x|^{1+2\gamma})\rangle$ computes like M_2 on input $\langle \phi(x), h'(b\,|x|^{1+2\gamma})\rangle$ with M_3 computing each bit of $\phi(x)$ as required by the simulation of M_2. Since it takes polylogarithmic time and logarithmic space to compute a bit of $\phi(x)$, the simulation of M_2 takes time $t(n) < (n^{1+2\gamma})^{\alpha(n^{1+2\gamma})} \log^k n$ and requires space $s(n) < (n^{1+2\gamma})^{1-\varepsilon}$. Since $\alpha(n) \in o(\log n)$, it follows that $t(n) \in n^{O(\alpha(n))}$, and moreover, since $\gamma = \varepsilon/4$, it follows that $s(n) \in O(n^{1-\delta})$ for some $\delta > 0$. Finally, $|h'(bn^{l+2\gamma})|$ is subpolynomial, and so,

$$L(M_1) \in \mathrm{NTISP}(n^{O(\alpha(n))}, n^{1-\delta})/n^{o(1)}.$$

Hence, $L \in \mathrm{NTISP}(n^{O(\alpha(n))}, n^{1-\delta})/n^{o(1)}$. But by Lemma 4.2 the latter class is in $\Sigma^n_{O(\alpha(n))}/n^{o(1)}$, a contradiction. ▌

5. EXTENDING THE LIPTON–VIGLAS LOWER BOUNDS

The main result proved in this section is the following theorem:

THEOREM 5.1. *Let α, r, and ε be rational numbers such that $\alpha \geqslant 1$, $r \geqslant 1$, $0 < \varepsilon \leqslant \frac{1}{r}$ and $\alpha + \varepsilon < \frac{r+1}{r}$. Suppose furthermore that,*

$$\mathrm{NTIME}(n^r) \subseteq \mathrm{DTISP}(n^{\alpha r}, n^{\varepsilon r})/o(n^{\frac{\alpha+\varepsilon}{r+1}}).$$

Then $\alpha \geqslant \frac{1}{2}(\sqrt{\varepsilon^2 + 4 + 4/r} - \varepsilon)$.

In particular, Theorem 5.1 implies the following time–space tradeoff for linear nondeterministic time:

THEOREM 5.2. *Suppose that*

$$\mathrm{NTIME}(n) \subseteq \mathrm{DTISP}(n^{\alpha}, n^{\varepsilon})/o(n^{\frac{1}{2}(\alpha+\varepsilon)}),$$

where $0 < \varepsilon \leqslant 1$, $\alpha \geqslant 1$ *and* $\alpha + \varepsilon < 2$. *Then,* $\alpha \geqslant \frac{1}{2}(\sqrt{\varepsilon^2 + 8} - \varepsilon)$.

In conjunction with Lemma 2.3, Theorem 5.2 gives time–space lower bounds for **SAT** on nonuniform machines.

COROLLARY 5.1. *For all* ε, $0 < \varepsilon < 1$, *and all* $\alpha \geqslant 1$ *such that* $\alpha + \varepsilon < 2$, *if*

$$\mathbf{SAT} \in \mathrm{DTISP}(n^{\alpha}, n^{\varepsilon})/o(n^{d}),$$

where $d < \frac{1}{2}(\alpha+\varepsilon)$, *then* $\alpha \geqslant \frac{1}{2}(\sqrt{\varepsilon^2+8}-\varepsilon)$. *More concretely, for all* ε, $0 < e < 1$, *and all* $\delta > 0$,

$$\mathbf{SAT} \notin \mathrm{DTISP}(n^{\frac{1}{2}(\sqrt{\varepsilon^2+8}-\varepsilon)-\delta}, n^{\varepsilon})/o(n^{\frac{1}{4}(\sqrt{\varepsilon^2+8}+\varepsilon)-\delta}).$$

Proof. Fix ε and assume the hypotheses. Then there exist a random access deterministic machine M running in time $O(n^{\alpha})$ and space $O(n^{\varepsilon})$ and a function $h\colon \mathbb{N} \to \Sigma^*$ such that $|h| \in o(n^d)$ and x is in **SAT** iff M accepts $\langle x, h(|x|)\rangle$.

Let $L \in \mathrm{NTIME}(n)$. Lemma 2.3 implies that for each x, $|x| = n$, there exists a propositional formula $\phi(x)$ of length $bn \log^2 n$, for some constant b independent of x, such that $x \in L$ iff $\phi(x) \in \mathbf{SAT}$. Moreover, the ith bit of $\phi(x)$ can be computed in $O(\log^{O(1)} n)$ time and $O(\log n)$ space provided random access to x.

Consider a machine M' that on input $\langle x, h(b\,|x| \log^2 |x|)\rangle$ computes like M on $\langle \phi(x), h(b\,|x| \log^2 |x|)\rangle$ where M' computes each bit of $\phi(x)$ as required by the simulation of M. The simulation takes $n^{\alpha+o(1)}$ time and $n^{\varepsilon+o(1)}$ space. Moreover, $x \in L$ iff M' accepts $\langle x, h(b\,|x| \log^2 |x|)\rangle$. Hence, $L \in \mathrm{DTISP}(n^{\alpha+o(1)}, n^{\varepsilon+o(1)})/o(n^{d+o(1)})$. But $L \in \mathrm{NTIME}(n)$ is arbitrary, and hence all of $\mathrm{NTIME}(n)$ is in the above class. Theorem 5.2 gives the lower bound for α. ∎

In particular, Corollary 5.1 implies that for all $\delta > 0$,

$$\mathbf{SAT} \notin \mathrm{DTISP}(n^{1+o(1)}, n^{1-\delta})/n^{1-\delta},$$
$$\mathbf{SAT} \notin \mathrm{DTISP}(n^{1.186}, \sqrt{n})/n^{0.843},$$
$$\mathbf{SAT} \notin \mathrm{DTISP}(n^{\sqrt{2}-\delta}, n^{o(1)})/o\left(n^{\frac{\sqrt{2}}{2}-\delta}\right).$$

It seems likely that separating $\mathrm{DTIME}(n^k)$ and $\mathrm{NTIME}(n^k)$ should be easier than separating P and NP. Indeed, using different techniques than those used in this paper, Paul *et al.* [24] were able to separate $\mathrm{DTIME}(n)$ and $\mathrm{NTIME}(n)$ for multi-tape Turing machines. The following weaker separation in the general case is an immediate corollary of Theorem 5.1:

Theorem 5.3. *For all rational* $r \geqslant 1$ *and all* $\varepsilon > 0$,

$$\mathrm{DTISP}(n^r, n^{1-\varepsilon}) \subsetneqq \mathrm{NTIME}(n^r).$$

Note that when $r = 1$, Theorem 5.3 only says that $\mathrm{DTISP}(n, n^{1-\varepsilon})$ is properly contained in $\mathrm{NTIME}(n)$ and thus appears weaker than the Paul *et al.* result; however, Theorem 5.3 holds for random access multitape Turing machines, whereas the separation from [24] only holds for standard multitape Turing machines.

5.1. Proof of Theorem 5.1

The arguments used to prove Theorem 5.1 are based on the arguments used by Kannan [18] and Lipton and Viglas [21]. As in those two papers, a translation lemma is needed.

LEMMA 5.1. *If* $\mathrm{coNTIME}(n^r) \subseteq \mathrm{NTISP}(n^{\alpha r}, n^{\beta r})/a(n)$ *for some rational numbers* $r \geqslant 1$, $\alpha \geqslant 1$, *and* $\beta > 0$, *and some function* $a(n)$, *then for any rational number* $q > r$,

$$\mathrm{coNTIME}(n^q) \subseteq \mathrm{NTISP}(n^{\alpha q} \log n, n^{\beta q})/a(n^{q/r}).$$

The proof of the lemma requires the following basic proposition also used in the proof of Theorem 5.1. Similar results appear in [18].

PROPOSITION 5.1. *For all rational numbers* $r > 0$, n^r *(in binary) is constructible from input of length* n *in* $O(n)$ *time and logarithmic auxiliary space.*

Proof (Proof of Lemma 5.1). Suppose the hypotheses and fix $L \in \mathrm{coNTIME}(n^q)$. Consider $L' = \{x10^{f(|x|)} : x \in L\}$ where $f(n) = n^{q/r}$. Then $L' \in \mathrm{coNTIME}(n^r)$ and hence $L' \in \mathrm{NTISP}(n^{\alpha r}, n^{\beta r})/a(n)$. So there exist a Turing machine M that runs in $n^{\alpha r}$ time and $n^{\beta r}$ space and a function $h : \mathbb{N} \to \Sigma^*$ such that $g = |h|$ is time-constructible, $g \leqslant a$ point-wise, and $x \in L$ iff M accepts $\langle x, h(|x|)\rangle$.

Consider a machine M' that on input $\langle x, y\rangle$ of length n computes as follows:

1. Compute $|x|$, $|y|$, $f(|x|)$, and $g(f(|x|))$, and write $f(|x|)$ onto an extra tape. Check that $|y| = g(f(|x|))$.
2. Simulate M on $z = \langle x10^{f(|x|)}, y\rangle$ and accept iff M accepts. Note that M' does not have space to write z. Instead, it uses an extra tape to write down in binary the position of M's input head and thus determine the character being read in the simulation of M. This incurs a logarithmic factor overhead for the simulation.

Since g is time-constructible, the first step can be done in linear time and logarithmic space with the aid of Proposition 5.1. The simulation step needs time $O(f(n)^{\alpha r} \log n)$, which is $O(n^{\alpha q} \log n)$, and space $O(f(n)^{\beta r})$, which is $O(n^{\beta q})$. ▮

Note that the extra log factor appearing in the conclusion of the above lemma can be eliminated; however, the extra precision is not required to prove Theorem 5.1.

Proof (Proof of Theorem 5.1). Assume that

$$\mathrm{NTIME}(n^r) \subseteq \mathrm{DTISP}(n^{\alpha r}, n^{\varepsilon r})/o(n^d), \tag{4}$$

where $d \leqslant \frac{\alpha+\varepsilon}{r+1}$. Note that $d < \frac{1}{r} \leqslant 1$. The assumption will be used to show that for some rational number c any co-nondeterministic machine running in time n^c can be simulated by a nondeterministic machine running in time $n^{(r/(r+1))\,c\alpha(\alpha+\varepsilon)+o(1)}$ and taking $o(n)$ advice. Theorem 3.1 then gives the desired lower bound for α.

Let $c = \frac{r+1}{\alpha+\varepsilon}$ and let $L \in \mathrm{coNTIME}(n^c)$ be arbitrary. Lemma 5.1 implies the existence of a deterministic Turing machine M running in time $O(n^{c\alpha} \log n)$ and space $O(n^{c\varepsilon})$ and a function $h\colon \mathbb{N} \to \Sigma^*$ such that $|h| \in o(n^{cd}) = o(n)$ and $x \in L$ iff M accepts $\langle x, h(|x|)\rangle$. Assume, without loss of generality, that there exist constants

a and b such that M takes exactly $an^{c\alpha}\log n$ steps on each path for all inputs of length n, and that all configurations of M on inputs of length n are $bn^{c\varepsilon}$ bits long.

Now, if $\beta < \alpha$, the computation of M can be simulated by a Σ_2 machine M' that existentially guesses the configurations $A_0, A_1, \ldots, A_{n^{c(\alpha-\beta)}}$ of M every $an^{c\beta}\log n$ steps, universally guesses an $i < n^{c(\alpha-\beta)}$, and then checks deterministically whether $A_i \vdash_M A_{i+1}$. By Eq. (4) and Lemma 5.1, the universal stage in the Σ_2 machine M' can be replaced by a deterministic computation taking sublinear advice, incurring a small time loss, to yield a nondeterministic machine M'' that takes sublinear advice and accepts $L(M)$. A nondeterministic machine taking sublinear advice that accepts L and runs in the same time as M'' can be constructed from M''. The goal is to pick β and construct M'' so that M'' runs as fast as possible.

To that end, begin by constructing a co-nondeterministic machine M_1 that given a string $A_0, \ldots, A_\ell$ of configurations of M, universally guesses an $i < \ell$ and checks that $A_i \vdash_M^* A_{i+1}$. In particular, let $\beta = \frac{r}{c}$ and consider a machine M_1 that on input w of length n computes as follows:

1. Check that w is of the form $x \# A_0 \# A_1 \# \cdots \# A_\ell$. Let $m = |x|$ and compute $bm^{c\varepsilon}$, $m^{c(\alpha-\beta)}$, and $am^{c\beta}\log m$. Check that $\ell = m^{c(\alpha-\beta)}$ and that each of the A_i has length $bm^{c\varepsilon}$.
2. Universally guess an $i < \ell$
3. Check that $A_i \vdash_M^{am^{c\beta}\log m} A_{i+1}$

Note that $\beta < \alpha$.

By Proposition 5.1, the first step takes linear time. The second step also takes linear time. Finally, the third step takes time $O(m^{c\beta}\log m)$ which is $O(n^r \log n)$. Equation (4) and Lemma 5.1 then imply the existence of a nondeterministic machine M_2 running in time $O(n^{\alpha r + o(1)})$ and a function $h' : \mathbb{N} \to \Sigma^*$ such that $|h'(n)| \in o(n^{d+o(1)})$ and $x \in L(M_1)$ iff M_2 accepts $\langle x, h'(|x|)\rangle$.

The nondeterministic machine M'' that accepts $L(M)$ with the aid of sublinear advice can now be defined. Since M'' will have to simulate M_2, it will require advice. The advice required on input of length n is $h'(f(n))$, where $f(n) = n + (1 + bn^{c\varepsilon})\, n^{c(\alpha-\beta)}$. Note that by the definitions of c and β, $f(n) \in O(n)$ and so, $|h'(f(n))| \in o(n^{d+o(1)}) \subseteq o(n)$. On input v of length n, M'' computes as follows:

1. Check that v is of the form $\langle x, y\rangle$. Let $m = |x|$. Check that $|y| = |h'(f(|x|))|$ and compute $bm^{c\varepsilon}$ and $m^{c(\alpha-\beta)}$.
2. Guess configurations $A_0, A_1, \ldots, A_{m^{c(\alpha-\beta)}}$ of M on input x each of length $bm^{c\varepsilon}$ where A_0 is the initial configuration of M on x and $A_{m^{c(\alpha-\beta)}}$ is an accepting configuration.
3. Let $z = x \# A_1 \# A_2 \# \cdots \# A_{m^{c(\alpha-\beta)}}$ and simulate M_2 on $w = \langle z, y\rangle$. Note that $|y| = |h'(|x|)|$.

By Proposition 5.1 the first step can be done in linear time and thus the computation of M'' is dominated by the third step. Since M_2 runs in $O(n^{\alpha r + o(1)})$ time and $|w| \in \Theta(|z|)$, the third step takes time $O(|z|^{\alpha + o(1)})$ which is $O(n^{(r/(r+1))\, c\alpha(\alpha+\varepsilon) + o(1)})$. So M'' runs in time $O(n^{(r/(r+1))\, c\alpha(\alpha+\varepsilon)+o(1)})$.

So $L(M) \in \mathrm{NTIME}(n^{(r/(r+1))\, c\alpha(\alpha+\varepsilon)+o(1)})/o(n)$. But, $x \in L$ iff M accepts $\langle x, h(|x|)\rangle$. Moreover, $|h(n)| \in o(n)$. Hence, $L \in \mathrm{NTIME}(n^{(r/(r+1))\, c\alpha(\alpha+\varepsilon)+o(1)})/o(n)$. As $L \in \mathrm{coNTIME}(n^c)$ is arbitrary, it follows that

$$\mathrm{coNTIME}(n^c) \subseteq \mathrm{NTIME}(n^{\frac{r}{(r+1)}\, c\alpha(\alpha+\varepsilon)+o(1)})/o(n).$$

Theorem 3.1 then implies that $\frac{r}{r+1}\,\alpha(\alpha+\varepsilon) \geqslant 1$. ∎

6. LOWER BOUNDS FOR CO-NONDETERMINISTIC TIME

The techniques used to prove Theorem 5.1 can also be used to obtain slightly weaker nonuniform nondeterministic time–space tradeoffs for co-nondeterministic time.

THEOREM 6.1. *Let α, r, and ε be rational numbers such that $\alpha \geqslant 1, r \geqslant 1$, $o < \varepsilon \leqslant \frac{\alpha}{r+\alpha-1}$, and $\frac{\alpha+\varepsilon}{\alpha+r} < \frac{1}{\alpha r}$. Let $c = \max(\frac{\alpha+r}{\alpha(\alpha+\varepsilon)}, \frac{1}{\varepsilon})$ and suppose furthermore that $(1 - c\,\frac{\alpha^2 - r\varepsilon}{\alpha+r})(\frac{\alpha+\varepsilon}{\alpha+r}) \leqslant \varepsilon$. If*

$$\mathrm{coNTIME}(n^r) \subseteq \mathrm{NTISP}(n^{\alpha r}, n^{\varepsilon r})/o(n^d),$$

where $d \leqslant \min(\frac{\alpha(\alpha+\varepsilon)}{\alpha+r}, \varepsilon)$, then $r\alpha^3 + \varepsilon r\alpha^2 - \alpha - r \geqslant 0$.

More concretely, by arguing as in the proof of Corollary 5.1, Theorem 6.1 gives the following lower bound for $\overline{\mathbf{SAT}}$:

COROLLARY 6.1. *For all ε, $0 < \varepsilon < 1$, if*

$$\overline{\mathbf{SAT}} \in \mathrm{NTISP}(n^\alpha, n^\varepsilon)/o(n^d),$$

where $d < \min(\frac{\alpha(\alpha+\varepsilon)}{\alpha+1}, \varepsilon)$, then $\alpha^3 + \varepsilon\alpha^2 - \alpha - 1 \geqslant 0$.

Corollary 6.1 gives the lower bounds for $\overline{\mathbf{SAT}}$ stated in the Introduction.

6.1. Proof of Theorem 6.1

The proof of Theorem 6.1 follows the same basic outline as the proof of Theorem 5.1. In addition, the following easy observation is required for the proof:

LEMMA 6.1. *If $\mathrm{coNTIME}(n^a) \subseteq \mathrm{NTIME}(n^b)/O(c(n))$ for some constants a and b and some advice function c such that $|c(n)| \in O(n)$, then*

$$\mathrm{coNTIME}(n^a)/O(c(n)) \subseteq \mathrm{NTIME}(n^b)/O(c(n)).$$

Here is a sketch of the proof for the lemma: suppose $L \in \mathrm{coNTIME}(n^a)/c(n)$ via some co-nondeterministic machine M that takes as advice some function h. Then $L(M) \in \mathrm{NTIME}(n^b)/c(n)$ via some nondeterministic machine M' that takes as advice some function h'. But then, L is accepted in time $O(n^b)$ by a nondeterministic machine that takes as advice both h and h' and that simulates M' on input $\langle x, h(|x|)\rangle$ and advice $h'(|\langle x, h(|x|)\rangle|)$.

Proof (Proof of Theorem 6.1). Fix $r \geqslant 1$ and $\varepsilon \leqslant \alpha/(r+\alpha-1)$ and assume that

$$\mathrm{coNTIME}(n^r) \subseteq \mathrm{NTISP}(n^{\alpha r}, n^{\varepsilon r})/o(n^d), \tag{5}$$

where $d \leqslant \min\left(\frac{\alpha(\alpha+\varepsilon))}{\alpha+r}, \varepsilon\right)$. Note that $d < 1/r \leqslant 1$. The assumption will be used to show that for some rational c any machine running in co-nondeterministic time n^c can be simulated by a nondeterministic machine running in time $n^{c\alpha^2 r((\alpha+\varepsilon)/(\alpha+r))+o(1)}$ and taking subpolynomial advice. Theorem 3.1 then gives the desired lower bound for α.

Let

$$c = \max\left(\frac{\alpha+r}{\alpha(\alpha+\varepsilon)}, \frac{1}{\varepsilon}\right),$$

and let $L \in \mathrm{coNTIME}(n^c)$ be arbitrary. In addition, assume that

$$\left(1 - \frac{c(\alpha^2 - r\varepsilon)}{\alpha+r}\right)\left(\frac{\alpha+\varepsilon}{\alpha+r}\right) \leqslant \varepsilon. \tag{6}$$

Note that $c \geqslant r$. So Eq. (5) and Lemma 5.1 imply the existence of a nondeterministic Turing machine M running in time $O(n^{c\alpha} \log n)$ and space $O(n^{c\varepsilon})$, and a function $h : \mathbb{N} \to \Sigma^*$ such that $|h(n)| \in o(n^{cd}) = o(n)$ and $x \in L$ iff M accepts $\langle x, h(|x|)\rangle$. Assume, without loss of generality, that there exist constants a and b such that M takes exactly $an^{c\alpha} \log n$ steps on each path for all inputs of length n, and that all configurations of M on input of length n can be described by $bn^{c\varepsilon}$ bits.

Now, if $\beta < \alpha$, the computation of M can be simulated by a Σ_3 machine M' that existentially guesses the configurations $A_0, A_1, \ldots, A_{n^{c(\alpha-\beta)}}$ of M every $n^{c\beta} \log n$ steps, universally guesses an $i < n^{c(\alpha-\beta)}$, and then guesses a valid computation path of M of length $n^{c\beta} \log n$ from A_i to A_{i+1}. By Eq. (5) and Lemma 5.1, the second "existential" stage in the Σ_3 machine M' can be replaced by a co-nondeterministic computation taking sublinear advice, incurring a small time loss, to yield a Σ_2 machine M'' that takes sublinear advice in its "universal" stage and accepts $L(M)$. Moreover, by Eq. (5) and Lemmas 5.1 and 6.1, the universal stage in M'' can be replaced by a nondeterministic computation taking sublinear advice, again incurring a small time loss, to yield a nondeterministic machine M''' that takes sublinear advice and accepts $L(M)$. A nondeterministic machine taking subpolynomial advice that accepts L and runs in the same time as M''' can then be constructed from M'''. The goal is to pick β and construct M''' so that M''' runs as fast as possible.

To that end, begin by constructing a nondeterministic machine M_1 that given two configurations A and B of M, existentially guesses a valid computation path from A to B. In particular, let $\beta = r\frac{\alpha+\varepsilon}{\alpha+r}$ and consider a machine M_1 that on input w of length n computes as follows:

1. Check that w is of the form $x \# A \# B$. Let $m = |x|$ and compute $bm^{c\varepsilon}$ and $m^{c\beta} \log m$. Check that both A and B are valid configurations of M each of length $bm^{c\varepsilon}$.

2. Existentially guess a valid computation path of M with input x from configuration A to B of length $m^{c\beta} \log m$.

The first step can be done in linear time with the aid of Proposition 5.1. The second step can be done in time $O(m^{c\varepsilon} + m^{c\beta} \log m)$. Since $c \geqslant 1/\varepsilon$, $n \in \Theta(m^{c\varepsilon})$ and M_1 runs in time $O(n + n^{\beta/\varepsilon} \log n)$. Moreover, since $\beta/\varepsilon \geqslant r \geqslant 1$, the latter is $O(n^{\beta/\varepsilon} \log n)$, and Eq. (5) and Lemma 5.1 imply that there exist a co-nondeterministic Turing machine M_2 running in time $O(n^{\alpha\beta/\varepsilon + o(1)})$ and a function h' mapping $\mathbb{N}$ to Σ^* such that $|h'(n)| \in o(n^{(d\beta)/(\varepsilon r)})$ and M_1 accepts w iff M_2 accepts $\langle x, h'(|w|)\rangle$.

Now construct a co-nondeterministic machine M_3 that, given a string $A_0, ..., A_\ell$ of configurations of M, universally guesses $i < \ell$ and checks that $A_i \vdash_M^* A_{i+1}$ by simulating M_2. On input w of length n, M_3 computes as follows:

1. Check that w is of the form $\langle x \# A_0 \# \cdots \# A_\ell, y\rangle$. Let $m = |x|$ and compute $bm^{c\varepsilon}$ and $am^{c(\alpha-\beta)}$. Check that $\ell = am^{c(\alpha-\beta)}$ and that each of the A_i has length $bm^{c\varepsilon}$. Check that $|y| = |h'(2 + m + 2bm^{c\varepsilon})|$.

2. Universally guess an $i < \ell$.

3. Simulate M_2 on input $\langle x \# A_i \# A_{i+1}, y\rangle$.

Note that M_3 requires advice for the simulation of M_2. The advice required on input of length n is $h'(f(n))$ where

$$f(n) = n \cdot \frac{n + 2(1 + bn^{c\varepsilon})}{n + (1 + bn^{c\varepsilon})\, an^{c(\alpha-\beta)}} \in O(n^{1-c(\alpha-\beta)}).$$

By Eq. (6) $|h'(f(n))| \in o(n^d)$ and hence the amount of advice required for the simulation is $o(n^d)$.

By Proposition 5.1, the first step in M_3's computation takes linear time. The second step also takes linear time. In the third step, M_2 is simulated on input of length $\Theta(m + m^{c\varepsilon})$ which is $\Theta(m^{c\varepsilon})$ since $c \geqslant 1/\varepsilon$. Since M_2 runs in $O(n^{\alpha\beta/\varepsilon + o(1)})$ time, the third step and hence the whole computation of M_3 takes time $O(m^{c\alpha\beta + o(1)}) \subseteq O(n^{r + o(1)})$. So the language L' consisting of those x for which M_3 accepts $\langle x, h'(f(|x|))\rangle$ is in $L' \in \text{coNTIME}(n^{r+o(1)})/o(n^d)$. Equation (5) and Lemmas 5.1 and 6.1 then imply the existence of a nondeterministic machine M_4 running in time $O(n^{\alpha r + o(1)})$ and a function h'': $\mathbb{M} \to \Sigma^*$ such that $|h''(n)| \in o(n^{d + o(1)})$ and $x \in L'$ iff M_4 accepts $\langle x, h''(|x|)\rangle$.

The nondeterministic machine M''' that accepts $L(M)$ with the aid of sublinear advice can now be defined. The advice M''' requires on input of length n is $h''(f'(n))$, where $f'(n) = n + (1 + bn^{c\varepsilon})\, an^{c(\alpha-\beta)}$. Note that by the definitions of c, d and β, it follows that $|h''(f'(n))| \in o(n)$. On input v of length n, M''' computes as follows:

1. Check that v is of the form $\langle x, y\rangle$. Let $m = |x|$. Check that $|y| = |h''(f'(|x|))|$ and compute $bm^{c\varepsilon}$ and $am^{c(\alpha-\beta)}$.

2. Guess $am^{c(\alpha-\beta)}$ configurations $A_0, ..., A_{am^{c(\alpha-\beta)}}$ each of length $bm^{c\varepsilon}$ of M on input x where A_0 is the initial configuration of M on x and $A_{am^{c(\alpha-\beta)}}$ is an accepting configuration.

3. Let $z = x\sharp A_1 \# A_2 \# \cdots \# A_{am^{c(\alpha-\beta)}}$ and simulate M_4 on $w = \langle z, y\rangle$. Note that $|y| = |h''(|z|)|$.

By Proposition 5.1, the first step can be done in linear time. The second step takes time $O(m^{c(\alpha-\beta+\varepsilon)})$ which is $O(n^{c(\alpha-\beta+\varepsilon)})$. Since M_4 runs in $O(n^{\alpha r+o(1)})$ time and $|w| \in \Theta(|z|)$, the third step takes time $O(|z|^{\alpha r+o(1)})$ which is $O(n^{c\alpha^2 r((\alpha+\varepsilon)/(\alpha+r))+o(1)})$. Hence, M''' runs in time $O(n^{c\alpha^2 r((\alpha+\varepsilon)/(\alpha+r))+o(1)})$.

Note that M accepts w iff M''' accepts $\langle w, h''(f'(|w|))\rangle$. Since $|h''(f'(n))| \in o(n)$, it follows that $L(M) \in \text{NTIME}(n^{c\alpha^2 r((\alpha+\varepsilon)/(\alpha+r))+o(1)})/o(n)$. But, $x \in L$ iff M accepts $\langle x, h(|x|)\rangle$. So $L \in \text{NTIME}(n^{c\alpha^2 r((\alpha+\varepsilon)/(\alpha+r))+o(1)})/o(n)$ and since $L \in \text{coNTIME}(n^c)$ is arbitrary, it follows that

$$\text{coNTIME}(n^c) \subseteq \text{NTIME}(n^{c\alpha^2 r \frac{\alpha+\varepsilon}{\alpha+r}+o(1)})/o(n).$$

Theorem 3.1 implies that $\alpha^2 r \frac{\alpha+\varepsilon}{\alpha+r} \geqslant 1$ and the theorem follows. ∎

7. DISCUSSION

In [11] Fortnow argued that simple diagonalization may yet be able to provide some new and interesting separations. By demonstrating that simple diagonalization can be used to obtain new separations of partially nonuniform and uniform classes, more ammunition has been given for Fortnow's thesis.

As noted in the Introduction, **SAT** is expected to require exponential deterministic time and exponential size circuits. Nevertheless, substantially improving the lower bounds presented in this paper, both uniform and nonuniform, seems quite difficult. Fortnow and van Melkebeek obtain better time–space tradeoffs in their paper [12], but the argument for the strongest tradeoff in their paper admits only subpolynomial advice rather than sublinear advice. Substantially strengthening the uniform lower bounds presented in this paper would resolve some long-standing open questions. For instance, if one could extend Theorem 5.1 to show that there exists an unbounded function $s(n)$ such that,

$$\text{NTIME}(n^k) \nsubseteq \text{DTISP}(n^{ks(k)}, \log^{O(1)} n),$$

for infinitely many $k \in \mathbb{N}$, then one would have a proof that SC $\neq$ NP. To see this, suppose to the contrary that SC = NP. Then for some integer ℓ, **SAT** is in DTISP(n^ℓ, $\log^{O(1)} n$). Lemma 2.3 then implies that

$$\text{NTIME}(n^j) \subseteq \text{DTISP}(n^{j\ell} \log^{O(1)} n, \log^{O(1)} n),$$

for all $j > \ell$, contradicting Eq. (7).

On the other hand, increasing the amount of nonuniformity in Theorem 3.1, and thereby possibly the amount of nonuniformity in all the nonuniform results presented in this paper, would also result in some major breakthroughs. Note that the factor limiting the amount of nonuniformity in the lower bounds for **SAT** proved in this paper is the limit on the amount of nonuniformity in Theorem 3.1. Indeed, increasing the amount of nonuniformity to $n^{1+o(1)}$ in Theorem 3.1 would allow one to show, using the arguments in the proof of Theorem 5.2, that **SAT** is not computable by a random access Turing machine taking $O(n^{1+o(1)})$ time, subpolynomial space, and $O(n \log n)$ bits of advice. But this would be a major breakthrough since the class of languages accepted by such machines includes the class of languages recognized by completely nonuniform linear size NC^1 circuits. To see this, note first that $O(n \log n)$ bits are enough to completely describe a linear size circuit. Now, design the encoding so that the (consecutive) bits in the encoding corresponding to a given gate describe (a) the function of the gate (needs $O(1)$ bits) and (b) the locations within the encoding of each of the gate's parents (needs $O(\log n)$ bits). Given such an encoding, a linear size log–depth circuit is evaluated using depth-first search. A stack of size $O(\log n)$ storing the addresses of the gates visited along the current path needs to be maintained resulting in a total space requirement of $O(\log^2 n)$. Note also that each gate is visited at most twice, and since we have random access, the processing time required whenever a gate is visited is polylogarithmic. It follows that the depth-first search can be done in $n^{1+o(1)}$ time and polylogarithmic space.

ACKNOWLEDGMENTS

I thank Valentine Kabanets, Steve Myers, Charlie Rackoff and, in particular, Steve Cook for many helpful discussions and comments. I thank Dieter van Melkebeek for many interesting discussions on this topic and for suggesting many improvements, including the use of random access machines.

REFERENCES

1. M. Ajtai, Determinism versus non-determinism for linear time RAMs with memory, *in* "Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing, Atlanta, GA, May 1–4, 1999," pp. 632–641, ACM Press.
2. M. Ajtai, A non-linear time lower bound for boolean branching programs, *in* "40th Annual Symposium on Foundations of Computer Science, New York, 17–19 Oct. 1999," pp. 459–464.
3. P. Beame, A general sequential time-space trade-off for finding unique elements, *SIAM J. Comput.* **20** (1991), 207–277.
4. P. Beame, M. Saks, X. Sun, and E. Vee, Super-linear time-space tradeoff lower bounds for randomized computation, *in* "FOCS2000," 2000.
5. P. Beame, M. Saks, and J. S. Thathachar, Time-space trade-offs for branching programs, *in* "FOCS: IEEE Symposium on Foundations of Computer Science (FOCS)," pp. 254–263, 1998.
6. R. V. Book, S. A. Greibach, and B. Wegbreit, Time- and tape-bounded Turing acceptors and AFLs, *J. Comput. System Sc.* **4** (1970), 606–621.
7. A. Cobham, The recognition problem for the set of perfect squares, *in* "Conference Record of the Seventh Annual Symposium on Switching and Automata Theory," pp. 78–87, IEEE, Press, New York, 1966.

8. S. A. Cook, The complexity of theorem-proving procedures, *in* "Proceedings of the 3rd ACM Symposium on the Theory of Computating," pp. 151–158, Assoc. Comput. Mach., New York, 1971.
9. S. A. Cook, Short propositional formulas represent nondeterministic computations, *Inform. Proces. Lett.* **26** (1998), 269–270.
10. P. Dúriś and Z. Galil, A time–space tradeoff for language recognition, *Math. Systems Theory* **17** (1984), 3–12.
11. L. Fortnow, Nondeterministic polynomial time versus nondeterministic logarithmic space: Time-space tradeoffs for satisfiability, *in* "Proceedings, Twelfth Annual IEEE Conference on Computational Complexity, Ulm, Germany, 24–27 June 1997," pp. 52–60, IEEE Computer Society Press, Los Alamitos, CA.
12. L. Fortnow and D. van Melkebeek, Time-space tradeoffs for nondeterministic computation, *in* "Proceedings, Fifteenth Annual IEEE Conference on Computational Complexity, Florence, Italy, 4–7 July 2000," pp. 2–13, IEEE Computer Society Press, Los Alamitos, CA.
13. S. Gupta, Alternating time versus deterministic time: A separation, *Math. Systems Theory,* **29** (1996), 661–672.
14. Y. Gurevich and S. Shelah, Nondeterministic linear-time tasks may require substantially nonlinear deterministic time in the case of sublinear workspace, *J. Assoc. Comput. Mach.* **37** (1990), 674–687.
15. F. C. Hennie and R. E. Stearns, Two-tape simulations of multitape Turing machines, *J. Assoc. Comput. Mach.* **13** (1966), 533–546.
16. J. Hopcroft, W. Paul, and L. Valiant, On time versus space, *J. Assoc. Comput. Mach.* **24** (1977), 332–337.
17. J. E. Hopcroft and J. D. Ullman, "Introduction to Automata Theory, Languages, and Computation," Addison–Wesley, Reading, MA, 1979.
18. R. Kannan, Towards separating nondeterminism from determinism, *Math. Systems Theory,* **17** (1984), 29–45.
19. M. Karchmer, Two time–space tradeoffs for element distinctness, *Theoret. Comput. Sc.* **47** (1986), 237–246.
20. R. M. Karp and R. J. Lipton, Some connections between nonuniform and uniform complexity classes, *in* "Conference Proceedings of the Twelfth Annual ACM Symposium on Theory of Computing, Los Angeles, CA, 28–30 Apr. 1980," pp. 302–309.
21. R. J. Lipton and A. Viglas, On the complexity of SAT, *in* "40th Annual Symposium on Foundations of Computer Science, New York, 17–19 Oct. 1999," pp. 459–464.
22. V. Nepomnjaščiĭ, Rudimentary predicates and Turing calculations, *Soviet Math. Dokl.* **11** (1970), 1462–1465.
23. C. Papadimitriou, "Computational Complexity," Addison–Wesley, Reading, MA, 1994.
24. W. J. Paul, N. Pippenger, E. Szemerédi, and W. T. Trotter, On determinism versus non-determinism and related problems (preliminary version), *in* "24th Annual Symposium on Foundations of Computer Science, Tucson, Arizona, 7–9 Nov. 1983," pp. 429–438.
25. N. Pippenger and M. J. Fischer, Relations among complexity measures, *J. Assoc. Comput. Mach.* **26** (1979), 361–381.
26. J. M. Robson, An $O(T \log T)$ reduction from RAM computations to satisfiability, *Theoret. Comput. Sci.* **82** (1991), 141–149.