Time-Space Lower Bounds for SAT on Uniform and Non-Uniform Machines

Iannis Tourlakis*
Department of Computer Science
University of Toronto
Toronto, Canada
iannis@cs.utoronto.ca

Abstract

The arguments used by Kannan [8], Fortnow [4], and Lipton-Viglas [10] are generalized and combined with a new argument for diagonalizing over machines taking n bits of advice on inputs of length n to obtain the first non-trivial time-space lower bounds for $\mathbf{SAT}$ *on non-uniform machines. In particular, we show that for any $a < \sqrt{2}$ and any $b < 1$,* $\mathbf{SAT}$ *cannot be computed by a random-access deterministic Turing machine using n^a time, $n^{o(1)}$ space and $n^{o(1)}$ advice, nor by a random-access deterministic Turing machine using $n^{1+o(1)}$ time, n^b space and $n^{o(1)}$ advice. More generally, we show that for all $\delta > 0$ and any $\epsilon < 1$,* $\mathbf{SAT}$ *cannot be solved by a random-access deterministic Turing machine using $n^{\frac{1}{2}(\sqrt{\epsilon^2+8}-\epsilon)-\delta}$ time, n^ϵ space and $n^{o(1)}$ advice. Similar lower bounds for computing* $\overline{\mathbf{SAT}}$ *on random-access nondeterministic Turing machines taking $n^{o(1)}$ advice are also obtained.*

Moreover, we show that $\mathbf{SAT}$ *does not have* NC^1 *circuits of size $n^{1+o(1)}$ 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 $\epsilon > 0$ and all rationals $r \geq 1$, $\mathsf{DTISP}(n^r, n^{1-\epsilon}) \subsetneq \mathsf{NTIME}(n^r)$. *We show how extending our uniform separations can lead to a separation of* SC *and* NP.

1 Introduction

The central question in complexity theory is in determining the relationship between P and NP. In [2] Cook showed that to separate these classes, it suffices to show that a deterministic machine requires superpolynomial time to recognize the set $\mathbf{SAT}$ of satisfiable propositional formulas. Consequently, obtaining lower bounds for $\mathbf{SAT}$ is one of the main goals of complexity theory. While it is expected that $\mathbf{SAT}$ requires exponential deterministic time and exponential size circuits, even superlinear lower bounds have proved elusive.

An important first step was made in [4] where Fortnow showed that $\mathbf{SAT}$ cannot be computed by a deterministic machine running in time $n^{1+o(1)}$ if the machine is also restricted to $n^{1-\epsilon}$ space for any $\epsilon > 0$. More generally, Fortnow showed that for all $\epsilon > 0$ and any unbounded function $r(n) \in O(\frac{\log n}{\log\log n})$,

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

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

Later, Lipton and Viglas [10], using techniques similar to the ones used by Fortnow, showed that for all $\delta > 0$, $\mathbf{SAT}$ cannot be computed by a deterministic Turing machine running in time $n^{\sqrt{2}-\delta}$ and polylogarithmic space. In addition, [10] claimed that $\mathbf{SAT} \notin \mathsf{DTISP}(n^{2-\delta}, \log^{O(1)} n)$ for all $\delta > 0$; however, this result was subsequently retracted [18].

Both the Fortnow and Lipton-Viglas results built on techniques used by Kannan [8] to show the existence of an integer k such that for all integers $j \neq k$, $\mathsf{NTIME}(n^j)$ properly contains $\mathsf{DTISP}(n^j, o(n^{j/k}))$. Indeed, all these results combine diagonalization arguments with work showing that if $\mathbf{SAT}$ 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.

1.1 Statement of main results

In this paper the arguments used by Kannan, Fortnow, and Lipton-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 new uniform and partially non-uniform time-space lower bounds for $\mathbf{SAT}$.

The new diagonalization argument is presented in section 3. In section 4, partially non-uniform extensions to the Fortnow-Buhrman bounds in [4] are obtained with the

* Supported in part by NSERC Fellowship PGSA-208465-1998

aid of the diagonal argument from section 3 and by adapting the proofs in [4] to the non-uniform 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 subsection 4.1, Buhrman's lower bound is generalized by showing that $\mathbf{SAT}$ does not have NC^1 circuits of size $n^{1+o(1)}$ generated by a nondeterministic log-space transducer taking subpolynomial advice. In subsection 4.2, Fortnow's lower bound is generalized by showing that for all unbounded functions $r(n) \in O(\frac{\log n}{\log \log n})$ and any $\epsilon > 0$,

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

In section 5 the proof framework in Lipton and Viglas [10] is adapted and combined with the diagonal result from section 3 to provide time-space lower bounds for nondeterministic computations on deterministic machines taking subpolynomial advice. In particular, a general theorem is proved showing that for all rational $r \geq 1$ and all ϵ, $0 < \epsilon \leq 1/r$, if

$$\mathsf{NTIME}(n^r) \subseteq \mathsf{DTISP}(n^{\alpha r}, n^{\epsilon r})/n^{o(1)}, \tag{1}$$

then $\alpha \geq \frac{1}{2}(\sqrt{\epsilon^2 + 4 + 4/r} - \epsilon)$. This implies that for all rational $r \geq 1$ and all $\epsilon > 0$,

$$\mathsf{DTISP}(n^r, n^{1-\epsilon}) \subsetneq \mathsf{NTIME}(n^r).$$

This result is similar to Kannan's result [8] mentioned above. However, as the constant in Kannan's result is unknown, the exact relationship with Kannan's result is uncertain. Equation (1) also implies that for all $\delta > 0$ and any ϵ, $0 < \epsilon < 1$, $\mathbf{SAT}$ cannot be computed by a random-access deterministic Turing machine using $n^{\frac{1}{2}(\sqrt{\epsilon^2+8}-\epsilon)-\delta}$ time, n^ϵ space and $n^{o(1)}$ advice. In particular, for all $\delta > 0$,

$$\mathbf{SAT} \notin \mathsf{DTISP}(n^{1.186}, \sqrt{n})/n^{o(1)},$$
$$\mathbf{SAT} \notin \mathsf{DTISP}(n^{\sqrt{2}-\delta}, n^{o(1)})/n^{o(1)}.$$

Time-space lower bounds for co-nondeterministic time on nondeterministic machines are obtained in section 5. A general theorem is given showing that for all rational $r \geq 1$ and all sufficiently small positive ϵ, if

$$\mathsf{coNTIME}(n^r) \subseteq \mathsf{NTISP}(n^{\alpha r}, n^{\epsilon r})/n^{o(1)},$$

then $r\alpha^3 + \epsilon r \alpha^2 - \alpha - r \geq 0$. The following lower bounds for $\overline{\mathbf{SAT}}$ are subsequently derived:

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

2 Preliminaries

The model of computation used is the random-access multi-tape Turing machine as defined in [5]. Such machines are like normal multi-tape Turing machines except that they have two types of tapes: index and non-index tapes. With the exception of the output tape, the tapes come in pairs. Each non-index 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 non-index 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 non-index 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 polylogorithmic losses in efficiency while still being easy to work with.

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 [7, 12].

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

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 $|\langle x, y \rangle| = 2|x| + |y| + 2$. As in Karp and Lipton [9], non-uniform classes will be defined using "advice functions". Let $\mathcal{C}$ be a class of languages and let $\mathcal{F}$ be a collection of functions mapping $\mathbb{N}$ to $\mathbb{N}$. A language L belongs to the (non-uniform) class $\mathcal{C}/\mathcal{F}$ if there exists a language L' in $\mathcal{C}$ and a function $h : \mathbb{N} \to \Sigma^*$ such that $|h| \in \mathcal{F}$ and $x \in L$ iff $\langle x, h(|x|) \rangle \in L'$. This paper will be primarily interested with non-uniform classes of the form $\mathcal{C}/n^{o(1)}$. Note that $f(n) \in n^{o(1)}$ iff $f(n^k) \in o(n)$ for all $k \in \mathbb{N}$.

Only time-constructible functions will be considered. A function f is time-constructible if there exists a $O(f(n))$ time-bounded deterministic (multi-tape) Turing machine that on inputs of length n outputs $f(n)$ in binary. Note that the function $\log n$ will be used to denote the integer-valued function $\lceil \log_2(n+1) \rceil$.

Let $\mathsf{NC}^1[\mathsf{NL}/n^{o(1)}]$ consist of all languages L for which there exist a family $\{C_n\}$ of NC^1 circuits accepting L, a nondeterministic log-space transducer M, and a function $h : \mathbb{N} \to \Sigma^*$ such that $|h| \in n^{o(1)}$ and M on input $\langle 1^n, h(n) \rangle$ outputs C_n. Note that $\mathsf{NC}^1[\mathsf{NL}/n^{o(1)}]$ 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 $\Sigma_{a(n)}^{t(n)}$ 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 $\Sigma_{a(n)}^{p}$. 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 $\overline{M}$ the machine that behaves just like M except that if M enters an accepting (rejecting) state, then $\overline{M}$ enters a rejecting (accepting) state, and if M enters an existential (universal) state, then $\overline{M}$ enters a universal (existential) state. Note that $L(\overline{M}) = \overline{L(M)}$. 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, $A \vdash_M^k B$ is true if M starting in A can reach B in k steps.

2.1 Reducing computations to SAT

In [3] Cook showed that a nondeterministic multi-tape Turing machine computation of time $t(n)$ can be reduced to a $\mathbf{SAT}$ question of size $O(t(n) \log t(n))$. Cook's result built on work by Hennie and Stearns [6] and Pippenger and Fischer [14] showing that a deterministic multi-tape 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 1 (Cook). *Let M be a $t(n)$ time-bounded alternating multi-tape Turing machine making a bounded number $a(n) \geq 1$ 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 $\mathbf{QBF}_{a(n)}$. 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 $\phi(x)$ in $O(\log^{O(1)} t(n))$ time and $O(\log t(n))$ space.*

In [16] 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 $\mathbf{SAT}$ 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 $\mathbf{SAT}$. For the results in this paper it will be necessary that the formulas $\phi(x)$ be easy to compute in the sense that a machine with random access to x can compute the ith bit of $\phi(x)$ using 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 parts of Robson's formulas not concerned 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 component of a formula in Robson's reduction has 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 multi-tape 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 1 then implies the existance of a function ϕ' mapping inputs y, $|y| = n$, to propositional formulas of size $O(n \log n)$ such that $\phi'(y)$ is in $\mathbf{SAT}$ 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. *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 $\mathbf{SAT}$. 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 3. *Let M be a $\Sigma_{a(n)}^{t(n)}$ machine where $a(n) \geq 1$ is bounded. 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 $\mathbf{QBF}_{a(n)}$. 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 3 is used in this paper to obtain lower bounds for $\mathbf{SAT}$. Moreover, the arguments in [10] seem to require Lemma 3 (or at least Lemma 1) to obtain lower bounds for $\mathbf{SAT}$. Indeed, Lipton and Viglas obtain time-space trade-offs for $\mathsf{NTIME}(n)$ in [10], but do not explicitly show how these imply a lower bound for $\mathbf{SAT}$. Lemma 3 can be used to provide the missing argument.

3 Diagonalization

The main new technical tool used in this paper is the following theorem together with variations of it. The proof refines an argument used by Rackoff [15] to construct a language in $\mathsf{DTIME}(s(n)\log s(n))$ that cannot be computed by circuits of size $s(n)$ having easily decodable descriptions of size n.

Theorem 4. *Let $t(n)$ and $T(n)$ be functions such that $t(3n+2) \in o(T(n))$. Then*

$$\mathsf{NTIME}(T(n)) \not\subseteq \mathsf{coNTIME}(t(n))/n.$$

Proof. Fix an enumeration of all co-nondeterministic 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 $\mathsf{NTIME}(T(n))$.

The claim is that L is not in $\mathsf{coNTIME}(t(n))/n$. 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 multi-tape Turing machine with only a constant factor slowdown [1], it can be shown that a random-access two-tape nondeterministic Turing machine can simulate a random-access multi-tape Turing machine with only a constant factor slowdown. Hence, U can simulate n steps of $\overline{M}$'s computation in cn steps where c only depends on M. Let m be a sufficiently long encoding of M such that $ct(3m+2) \le T(m)$. Consider U on input $h(m)$. By definition, U simulates $\overline{M_m} = \overline{M}$ on input $\langle h(m), h(m)\rangle$. Now recall that $|\langle x, y\rangle| = 2|x| + |y| + 2$, whence, $|\langle h(m), h(m)\rangle| = 3|h(m)| + 2 = 3m + 2$. So the simulation of $\overline{M}$ on $\langle h(m), h(m)\rangle$ requires time $ct(3m+2) \le T(m)$, and thus, U has enough time to complete the simulation. But then, $h(m)$ is in L iff U accepts $h(m)$ iff $\overline{M}$ 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 $\langle,\rangle$ and thus not too much importance should be attached to the precise constants appearing.

In the same way that the nondeterministic time hierarchy seems to be much harder to prove as compared with the deterministic time hierarchy, it seems to be harder to prove the existence of a language in $\mathsf{NTIME}(n^k) - (\mathsf{NTIME}(o(n^k))/n)$ than proving Theorem 4. This follows from the fact that the relationship between nondeterministic and co-nondeterministic time is poorly understood.

Note that the arguments in the proof of Theorem 4 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 $\Sigma^{n\log n}_{s(n)}$ is not contained in $\Sigma^{n}_{O(1)}/n$ when $s(n)$ is unbounded, a result that will be used in the proof of Theorem 5 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 $\mathbf{SAT}$:

Theorem 5. $\mathbf{SAT}$ *cannot both have circuits of size $n^{1+o(1)}$ and be in $\mathsf{NL}/n^{o(1)}$. In particular,* $\mathbf{SAT}$ *cannot be computed by $\mathsf{NC}^1[\mathsf{NL}/n^{o(1)}]$ circuits of size $n^{1+o(1)}$.*

The proof of Theorem 5 is similar to the proof given in [4] for Buhrman's original result. Like in Buhrman's original result, the proof relies on the following two lemmas. The first appeared in Fortnow [4] where it was credited to Buhrman. The second follows easily from the version of Nepomnjaščiĭ's Theorem [11] presented in Kannan [8] and later in Fortnow [4].

Lemma 6 (Buhrman). *Suppose* $\mathbf{SAT}$ *has circuits of size $n^{1+o(1)}$. Then $\mathbf{QBF}_{s(n)} \in \Sigma^p_2$ for some unbounded function $s(n) \in O(\frac{\log n}{\log\log n})$.*

Lemma 7 (Nepomnjaščiĭ's Theorem). *Let $\epsilon > 0$. Then for any functions $t(n)$ and $\alpha(n)$, $1 \le \alpha(n) \le n^{o(1)}$,*

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

The proof of Theorem 5 is by contradiction. The first hypothesis, in conjunction with Lemma 6, is used to show that $\Sigma^{n\log n}_{s(n)} \subseteq \Sigma^p_2$. The second hypothesis is then used to show that $\Sigma^p_2 \subseteq \mathsf{NL}/n^{o(1)}$. Lemma 7 then implies that $\Sigma^{n\log n}_{s(n)} \subseteq \Sigma^{n}_{O(1)}/n^{o(1)}$. Diagonalizing as in the proof of Theorem 4 then gives the desired contradiction.

Proof of Theorem 5. Assume $\mathbf{SAT}$ has circuits of size $n^{1+o(1)}$. Lemma 6 then implies that

$$\Sigma^{n\log n}_{s(n)} \subseteq \Sigma^p_2. \tag{2}$$

Now assume in addition that $\mathbf{SAT} \in \mathsf{NL}/n^{o(1)}$ to derive a contradiction. The first step is to show that this assumption implies that $\Sigma_2^p \subseteq \mathsf{NL}/n^{o(1)}$. To that end, note that any language in $\mathsf{NP}/n^{o(1)}$ can be reduced to $\mathbf{SAT}$ via a log-space transducer taking subpolynomial advice. So if $\mathbf{SAT} \in \mathsf{NL}/n^{o(1)}$, it follows that $\mathsf{NP}/n^{o(1)} = \mathsf{NL}/n^{o(1)}$. In particular, $\mathsf{NP}/n^{o(1)} = \mathsf{coNP}/n^{o(1)}$.

Now fix $A \in \Sigma_2^p$. There exists a polynomial f and a relation $R \in \mathsf{coNP}$ such that $x \in A$ iff there exists y, $|y| \leq f(|x|)$, such that $(x, y) \in R$ (assume that $|(x, y)| = |x| + |y|$). Moreover, by padding y (y can be thought of as the list of non-deterministic choices made in the "existential stage" of the Σ_2^p machine accepting A, and so, it can be padded with unused extra choices), assume without loss of generality that $|y| = f(|x|)$. Since $\mathsf{coNP}/n^{o(1)} = \mathsf{NP}/n^{o(1)}$, it follows that $R \in \mathsf{NP}/n^{o(1)}$. So there exist a polynomial time nondeterministic machine M and a function $h : \mathbb{N} \to \Sigma^*$ 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 $\langle (x, y), w \rangle$. Note that M' runs in polynomial time and that $x \in A$ iff M' accepts $\langle x, h(|x| + f(|x|)) \rangle$. Moreover, since $|h|$ is subpolynomial, $|h(n + f(n))|$ is also subpolynomial and thus, $A \in \mathsf{NP}/n^{o(1)} = \mathsf{NL}/n^{o(1)}$. So $\Sigma_2^p \subseteq \mathsf{NL}/n^{o(1)}$.

Equation (2) and Lemma 7 then imply that

$$\Sigma_{s(n)}^{n \log n} \subseteq \Sigma_2^p \subseteq \mathsf{NL}/n^{o(1)} \subseteq \Sigma_{O(1)}^{n}/n^{o(1)}.$$

A contradiction is now obtained by finding a language in $\Sigma_{s(n)}^{n \log n} - \Sigma_{O(1)}^{n}/n$ by generalizing the arguments in the proof of Theorem 4. $\square$

The proof of Theorem 5 can be adapted to show that $\mathbf{SAT}$ does not have circuits of quasilinear size and subpolynomial width 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 4, the following extension to Fortnow's time-space tradeoff for $\overline{\mathbf{SAT}}$ can be proved:

Theorem 8. *Let $r(n)$ be any unbounded function such that $r(n) \in O(\frac{\log n}{\log \log n})$. Then for all $\epsilon > 0$,*

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

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

$$\overline{\mathbf{SAT}} \notin \mathsf{NTISP}(n^{1+o(1)}, n^{1-\epsilon})/n^{o(1)}.$$

An alternate proof of the latter result appears in the next section.

The arguments used to prove Theorem 8 follow along the same lines as those used by Fortnow to prove his original result. The essential differences lies in the introduction of the new diagonal argument to extend to the non-uniform case. As in Fortnow, a version of Nepomnjaščiĭ's Theorem is required. The non-uniform version given by Lemma 7 will suffice. Moreover, a version of Fortnow's lemma giving sufficient conditions for collapsing a non-constant number of levels of the polynomial hierarchy is required:

Lemma 9. *Let $r(n) \in O(\frac{\log n}{\log \log n})$ be an unbounded function and suppose that $\overline{\mathbf{SAT}} \in \mathsf{NTIME}(n^{1+\frac{1}{r(n)}})/n^{o(1)}$. Then for any $\gamma > 0$, there exists a constant ℓ such that*

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

Before giving the proof for Lemma 9, it will be useful to recall the proof of its uniform equivalent in Fortnow [4]:

Lemma 10 (Fortnow). *Let $r(n) \in O(\frac{\log n}{\log \log n})$ be unbounded and suppose that $\overline{\mathbf{SAT}} \in \mathsf{NTIME}(n^{1+\frac{1}{r(n)}})$. Then for any $\gamma > 0$, there exists a constant ℓ such that*

$$\Sigma_{\frac{r(n)}{\ell}}^{n \log n} \subseteq \mathsf{coNTIME}(n^{1+\gamma}).$$

The argument behind Fortnow's lemma is as follows: fix $\gamma > 0$ and for simplicity, consider the case where $r(n) = \frac{\log n}{\log \log n}$ so that $n^{1+\frac{1}{r(n)}} = n \log n$. The goal is to show that if $\overline{\mathbf{SAT}} \in \mathsf{NTIME}(n \log n)$, then for some ℓ,

$$\Sigma_{\frac{\log n}{\ell \log \log n}}^{n \log n} \subseteq \mathsf{coNTIME}(n^{1+\gamma}).$$

Fortnow observes that if

$$A \in \Sigma_{\frac{\log n}{\ell \log \log n}}^{n \log n},$$

then

$$\bar{A} \in \Sigma_{\frac{\log n}{\ell' \log \log n}}^{n \log n},$$

for any $\ell' < \ell$. So it suffices to show

$$\Sigma_{\frac{\log n}{\ell \log \log n}}^{n \log n} \subseteq \mathsf{NTIME}(n^{1+\gamma}).$$

The idea is to use the fact that $\overline{\mathbf{SAT}} \in \mathsf{NTIME}(n \log n)$ to convert a propositional formula with $k-1$ quantifier alternations into a formula with $k-2$ alternations without substantially increasing the size of the formula. Hence, repeatedly applying the reduction gives a polynomial-size formula with no alternations equivalent to the original formula.

This is done as follows. Let M be a nondeterministic machine computing $\overline{\mathbf{SAT}}$ in $O(n \log n)$ time. Lemma 3 then implies that there exists a constant a and a function ϕ mapping inputs x, $|x| = n$, of M to propositional formulas of length $at(n) \log^4 t(n)$ such that M accepts x iff $\phi(x)$ is in $\mathbf{SAT}$. Moreover, ϕ is computable in $O(t(n) \log^4 t(n))$ time. Now let ψ_k be a formula of length m_k with $k-1$ alternations beginning with $\exists$. It suffices to consider the case where k is even as the case where k is odd is similar. So ψ_k is of the form

$$\psi_k = \exists \bar{x}_1 \forall \bar{x}_2 \dots \exists \bar{x}_{k-1} \forall \bar{x}_k \phi(\bar{x}_1, \dots, \bar{x}_k),$$

Consider the formula

$$\tau(\bar{x}_1, \dots, \bar{x}_{k-1}) = \forall \bar{x}_k \phi(\bar{x}_1, \dots, \bar{x}_k).$$

Note that the truth of $\tau(\bar{x}_1, \dots, \bar{x}_{k-1})$ is a $\overline{\mathbf{SAT}}$ question of length bounded by m_k. So, the truth of $\tau(\bar{x}_1, \dots, \bar{x}_{k-1})$ is equivalent to the truth of

$$\phi(\tau(\bar{x}_1, \dots, \bar{x}_{k-1})) = \exists \bar{y} \sigma(\bar{x}_1, \dots, \bar{x}_{k-1}, \bar{y}),$$

where the size of σ is bounded by $am_k \log^4 m_k$. So ψ_k is equivalent to

$$\psi_{k-1} = \exists \bar{x}_1 \forall \bar{x}_2 \dots \exists \bar{x}_{k-1} \exists \bar{y} \sigma(\bar{x}_1, \dots, \bar{x}_{k-1}, \bar{y}),$$

a formula with $k-2$ alternations and length m_{k-1} bounded by $am_k \log^4 m_k$. Pad the encoding of ψ_{k-1} to ensure that $m_{k-1} = am_k \log^4 m_k$. In particular, this ensures when reducing a formula ψ, the lengths of the intermediate formulas (i.e., the m_i) depend only on the length of ψ. This will be important in the proof of the non-uniform version Lemma 9.

Let $s(n) = \frac{\log n}{\ell \log \log n}$. By induction, Fortnow shows that for $\ell > \frac{16}{\gamma}$, a formula of length $n \log^4 n$ with $s(n)$ alternations can be reduced by the above method to a satisfiability question of length $m_1 = n^{1+\frac{\gamma}{2}}$.

Now, if $A \in \Sigma_{s(n)}^{n \log n}$, Lemma 1 says that asking if $x \in A$ is equivalent to asking if an $s(n) - 1$ alternation formula $\psi_{s(n)}$ of length $n \log^4 n$ is true. But by the above the latter is equivalent to asking whether a formula with no alternations of length $n^{1+\frac{\gamma}{2}}$ is in $\mathbf{SAT}$, which is certainly in $\mathsf{NTIME}(n^{1+\gamma})$.

A sketch of the proof of Lemma 9 can now be given. Note first that as opposed to Fortnow's lemma, the assumption is now that $\overline{\mathbf{SAT}} \in \mathsf{NTIME}(n \log n)/n^{o(1)}$. So instead of there being a nondeterministic machine solving $\overline{\mathbf{SAT}}$ in $O(n \log n)$ time, there exist a nondeterministic machine M running in time $O(n \log n)$ and a function $h : \mathbb{N} \to \Sigma^*$ such that $|h|$ is subpolynomial and $x \in \overline{\mathbf{SAT}}$ iff M accepts $\langle x, h(|x|)\rangle$. Hence, for a machine M' to do the reductions required in the proof of Fortnow's lemma, it requires on inputs of size n to have the subpolynomial advice required by M to solve $\overline{\mathbf{SAT}}$ on instances of size $m_{s(n)}, m_{s(n)-1}, \dots, m_2$, and m_1, where $s(n)$ is as defined above, $m_{s(n)} = n \log^4 n$ and $m_{k-1} = am_k \log^4 m_k$, $1 < k \le s(n)$. Since $m_1 \in n^{O(1)}$ and $s(n) \in O(\log n)$, the advice required is the product of two subpolynomial functions. But the product of two subpolynomial functions is also subpolynomial, and hence, the advice required by M' is subpolynomial. Lemma 9 now follows.

Proof of Theorem 8. Fix $r(n)$ and ϵ, and let ℓ be the constant guaranteed by Lemma 9 for $\gamma = \epsilon/4$. Let $\alpha(n)$ be any function such that $\alpha(n) \in o(r(n))$. Construct a language L in $\Sigma_{\frac{r(n)}{\ell}}^{n \log n} - (\Sigma_{O(\alpha(n))}^{n}/n^{o(1)})$ by arguing as in the proof of Theorem 4. Now assume that

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

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

Since $\overline{\mathbf{SAT}} \in \mathsf{NTIME}(n^{1+\frac{1}{r(n)}})$, Lemma 9 implies that $\overline{L}$ is in $\mathsf{NTIME}(n^{1+\gamma})/n^{o(1)}$. 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 $|h|$ is subpolynomial and x is in L iff M_1 accepts $\langle x, h(|x|)\rangle$. Note that $n^{1+\gamma} \log n^{1+\gamma} \in O(n^{1+2\gamma})$. By Lemma 3, there exists a constant b such that for each input x, $|x| = n$, there exists a CNF formula $\phi(x)$ of length $bn^{1+2\gamma}$ such that x is in $L(M_1)$ iff $\phi(x)$ is satisfiable. Moreover, there exists a k, Moreover, there exists a k, independent of n, such that the ith bit of $\phi(x)$ can be computed in time $O(\log^k n)$ and $O(\log n)$ space provided that there is random-access to x.

Since $\overline{\mathbf{SAT}} \in \mathsf{NTISP}(n^{\alpha(n)}, n^{1-\epsilon})/n^{o(1)}$, there exist a non-deterministic machine M_2 running in $n^{\alpha(n)}$ time and using $n^{1-\epsilon}$ space and a function $h' : \mathbb{N} \to \Sigma^*$ such that $|h'|$ is subpolynomial and x is in $\overline{\mathbf{SAT}}$ iff M_2 accepts $\langle x, h(|x|)\rangle$. This fact together with the properties enjoyed by ϕ will be used to construct a nondeterministic machine that accepts $\overline{\mathbf{SAT}}$ in $n^{1+o(1)}$ time and $n^{1-\epsilon}$ space.

To that end, consider a nondeterministic machine M_3 that computes as follows on input $\langle x, h'(b|x|^{1+2\gamma})\rangle$: M_3 computes like M_2 on input $\langle \phi(x), h'(b|x|^{1+2\gamma}\rangle$. However, M_3 does not actually write down $\phi(x)$; instead, it computes each bit of $\phi(x)$ as required by the simulation of M_2. As discussed above, it takes $O(\log^k n)$ time to compute a bit of $\phi(x)$. Hence, the simulation of M_2 by M_3 takes time $t(n) < (n^{1+2\gamma})^{\alpha(n^{1+2\gamma})} \log^k n$ and requires space $s(n) < (n^{1+2\gamma})^{1-\epsilon}$. Note that since $\alpha(n) \in o(\log n)$, it follows that $t(n) \in n^{O(\alpha(n))}$. Moreover, since $\gamma = \epsilon/4$, it follows that $s(n) \in O(n^{1-\delta})$ for some $\delta > 0$. Finally, $|h'(bn^{1+2\gamma})|$ is subpolynomial, and so,

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

It is easy to see that this implies that in fact L is in $\mathsf{NTISP}(n^{O(\alpha(n))}, n^{1-\delta})/n^{o(1)}$.

Lemma 7 then implies that $L \in \Sigma^n_{O(\alpha(n))}/n^{o(1)}$, a contradiction. $\square$

5 Extending the Lipton-Viglas lower bounds

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

Theorem 11. *For all rational numbers r and ϵ such that $r \geq 1$ and $0 < \epsilon \leq 1/r$, if*

$$\mathsf{NTIME}(n^r) \subseteq \mathsf{DTISP}(n^{\alpha r}, n^{\epsilon r})/n^{o(1)},$$

for some rational number $\alpha \geq 1$, then

$$\alpha \geq \frac{1}{2}\left(\sqrt{\epsilon^2 + 4 + 4/r} - \epsilon\right).$$

In conjunction with Lemma 3, Theorem 11 can be used to obtain time-space lower bounds for $\mathbf{SAT}$ on non-uniform machines.

Corollary 12. *For all ϵ, $0 < \epsilon < 1$, if*

$$\mathbf{SAT} \in \mathsf{DTISP}(n^{\alpha}, n^{\epsilon})/n^{o(1)},$$

then $\alpha \geq \frac{1}{2}\left(\sqrt{\epsilon^2 + 8} - \epsilon\right)$.

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

Let L be in $\mathsf{NTIME}(n)$. Lemma 3 implies the existence of a constant b such that for each input x, $|x| = n$, there exists a propositional formula $\phi(x)$ of length $bn\log^2 n$ such that x is in L iff $\phi(x)$ is satisfiable. Moreover, the ith bit of $\phi(x)$ can be computed in $O(\log^{O(1)} n)$ time and $O(\log n)$ space provided there is random-access to x.

Now consider a deterministic machine M' that on input $\langle x, h(b|x|\log^2|x|)\rangle$ computes as follows: M' computes like M on input $\langle \phi(x), h(b|x|\log^2|x|)\rangle$. However, M' does not actually write $\phi(x)$; instead, it computes each bit of $\phi(x)$ as required by the simulation of M. By the above, it takes $O(\log^{O(1)} n)$ time to compute a bit of $\phi(x)$. Hence, the simulation of M by M' thereby takes time $n^{\alpha}\log^{O(1)} n$ and requires $n^{\epsilon}\log^{O(1)} n$ space. Note moreover, that $x \in L$ iff M' accepts $\langle x, h(b|x|\log^2|x|)\rangle$. Hence, $L \in \mathsf{DTISP}(n^{\alpha+o(1)}, n^{\epsilon+o(1)})/n^{o(1)}$. Since L in $\mathsf{NTIME}(n)$ is arbitrary, it follows that

$$\mathsf{NTIME}(n) \subseteq \mathsf{DTISP}(n^{\alpha+o(1)}, n^{\epsilon+o(1)})/n^{o(1)}.$$

Theorem 11 then implies the desired lower bound for α. $\square$

Separating $\mathsf{DTIME}(n^k)$ and $\mathsf{NTIME}(n^k)$ should likely be less difficult than separating P and NP. Indeed, using different techniques than those used in this paper, Paul, Pippenger, Szemerédi and Trotter [13] were able to separate $\mathsf{DTIME}(n)$ and $\mathsf{NTIME}(n)$ for multi-tape Turing machines. Theorem 11 implies the following weaker separation in the general case:

Theorem 13. *For all rational $r \geq 1$ and all $\epsilon > 0$,*

$$\mathsf{DTISP}(n^r, n^{1-\epsilon}) \subsetneq \mathsf{NTIME}(n^r).$$

Note that when $r = 1$, Theorem 13 only says that $\mathsf{DTISP}(n, n^{1-\epsilon})$ is properly contained in $\mathsf{NTIME}(n)$ and thus appears weaker than the Paul, Pippenger, Szemerédi and Trotter result; however, Theorem 13 holds for random-access multi-tape Turing machines, whereas the separation from [13] only holds for multi-tape Turing machines.

The techniques used to prove Theorem 11 can also be used to obtain slightly weaker non-uniform nondeterministic time-space tradeoffs for co-nondeterministic time.

Theorem 14. *Let α, r and ϵ be rational numbers such that $\alpha \geq 1$, $r \geq 1$ and $0 < \epsilon \leq \alpha/(r + \alpha - 1)$. Suppose furthermore that*

$$\mathsf{coNTIME}(n^r) \subseteq \mathsf{NTISP}(n^{\alpha r}, n^{\epsilon r})/n^{o(1)}.$$

Then $r\alpha^3 + \epsilon r\alpha^2 - \alpha - r \geq 0$.

By arguing as in the proof of Corollary 12, Theorem 14 can be used to obtain the lower bounds for $\overline{\mathbf{SAT}}$.

Corollary 15. *For all ϵ, $0 < \epsilon < 1$, if*

$$\overline{\mathbf{SAT}} \in \mathsf{NTISP}(n^{\alpha}, n^{\epsilon})/n^{o(1)},$$

then $\alpha^3 + \epsilon\alpha^2 - \alpha - 1 \geq 0$.

Note that Corollary 15 implies Theorem 8, the non-uniform extension of Fortnow's [4] time-space tradeoff for $\overline{\mathbf{SAT}}$ from section 4.

Fortnow [4] showed that if M is a deterministic Turing machine accepting $\mathbf{SAT}$ in time $t(n)$ and space $s(n)$, then for some $\epsilon > 0$, $t(n)s(n) \in \Omega(n^{1+\epsilon})$. Corollary 12 can be used to give an improved time-space product for $\mathbf{SAT}$. The proof is similar to the one used by Fortnow to prove his time-space product.

Theorem 16. *Let M be a deterministic Turing machine deciding $\mathbf{SAT}$ using $t(n)$ time, $s(n)$ space, and subpolynomial advice. Then $t(n)s(n) \in \Omega(n^{(\sqrt{17}+1)/4})$. In particular, $t(n)s(n) \in \omega(n^{1.28})$.*

Proof. Let $0 < \epsilon < 1$, and suppose $s(n) \in \omega(n^{\epsilon})$. Then $t(n)s(n) \in \omega(n^{1+\epsilon})$ since M requires linear time to read its input. If on the other hand $s(n) \in O(n^{\epsilon})$, then $t(n)s(n) \in \Omega(n^{\frac{1}{2}(\sqrt{\epsilon^2+8}-\epsilon)})$ by Corollary 12. Hence, $t(n)s(n) \in \Omega(\min(n^{1+\epsilon}, n^{\frac{1}{2}(\sqrt{\epsilon^2+8}-\epsilon)}))$.

Now note that as ϵ goes from 0 to 1, $n^{1+\epsilon}$ increases while $n^{\frac{1}{2}(\sqrt{\epsilon^2+8}-\epsilon)}$ decreases. Hence, the largest product is obtained when the exponents of the latter two functions are equal, that is, when $2 + 2\epsilon = \sqrt{\epsilon^2+8} - \epsilon$. Hence, $\epsilon = (\sqrt{17}-3)/4$, whence $t(n)s(n) \in \Omega(n^{(\sqrt{17}+1)/4})$. $\square$

5.1 Proof of Theorem 11

The arguments used to prove Theorem 11 are very closely based on the arguments used by Kannan [8] and Lipton and Viglas [10]. As in those two papers, a translation lemma is needed.

Lemma 17. *If* $\mathsf{coNTIME}(n^r) \subseteq \mathsf{NTISP}(n^{\alpha r}, n^{\beta r})/n^{o(1)}$ *for some rational numbers* $r \geq 1$*,* $\alpha \geq 1$ *and* $\beta > 0$*, then for any rational number* $q > r$*,*

$$\mathsf{coNTIME}(n^q) \subseteq \mathsf{NTISP}(n^{\alpha q}\log n, n^{\beta q})/n^{o(1)}.$$

The proof of the lemma is contained in the appendix. The proof requires the following basic proposition which will also be used in the proof of Theorem 11. Similar results appear in [8].

Proposition 18. *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 of Theorem 11. Fix $r \geq 1$ and $\epsilon \leq 1/r$ and assume that

$$\mathsf{NTIME}(n^r) \subseteq \mathsf{DTISP}(n^{\alpha r}, n^{\epsilon r})/n^{o(1)}. \tag{3}$$

This assumption will be used to show that for some rational c any co-nondeterministic machine running in time n^c can be simulated by a nondeterministic machine running in time $n^{c\alpha r\frac{\alpha+\epsilon}{r+1}+o(1)}$ and taking subpolynomial advice. This fact coupled with Theorem 4 then gives the desired lower bound for α.

Let

$$c = \max(1/(\alpha - \beta + \epsilon), r),$$

where $\beta = \frac{r}{r+1}(\alpha + \epsilon)$. Note that $\beta = r(\alpha - \beta + \epsilon)$. Let $L \in \mathsf{coNTIME}(n^c)$ be arbitrary. Now, since L is in $\mathsf{coNTIME}(n^c)$, Equation (3) and Lemma 17 imply the existence of a deterministic Turing machine M running in time $an^{c\alpha}\log n$ (for some constant a) and space $O(n^{c\epsilon})$, and a function $h : \mathbb{N} \to \Sigma^*$ such that $|h|$ is subpolynomial and $x \in L$ iff M accepts $\langle x, h(|x|)\rangle$. In addition assume, without loss of generality, that M takes exactly $an^{c\alpha}\log n$ steps on each path for all inputs of length n. Moreover, assume without loss of generality, that there exists a constant b such that all configurations of M on input of length n can be described by $bn^{c\epsilon}$ bits.

Note that 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 $n^{c\beta}\log n$ steps, universally guesses an $i < n^{c(\alpha-\beta)}$, and then checks deterministically whether $A_i \vdash_M A_{i+1}$. By Equation (3) and Lemma 17, the "universal" stage in the Σ_2 machine M' can be replaced by a deterministic computation taking subpolynomial advice, incurring a small time loss, to yield a nondeterministic machine M'' that takes subpolynomial 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''. So the goal is to construct M'' so that it 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, 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 $bm^{c\epsilon}$, $m^{c(\alpha-\beta)}$ and $am^{c\beta}\log m$. Check that $\ell = m^{c(\alpha-\beta)}$ and that each of the A_i has length $bm^{c\epsilon}$.
2. Universally guess an $i < \ell$.
3. Check that $A_i \vdash_M^{am^{c\beta}\log m} A_{i+1}$

The first step can be done in linear time with the aid of Proposition 18. The second step also takes linear time. Finally, the third step takes time $O(m^{c\beta}\log m)$ which is $O(m^{cr(\alpha-\beta+\epsilon)}\log m)$. Note that $n \in \Theta(m + m^{c(\alpha-\beta+\epsilon)})$. But $c \geq 1/(\alpha - \beta + \epsilon)$, and so $n \in \Theta(m^{c(\alpha-\beta+\epsilon)})$. Hence, the third step takes time $O(n^r \log n)$, and M_1 runs in time $O(n^r \log n)$. Equation (3) and Lemma 17 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 $|h'|$ is subpolynomial and $x \in L(M_1)$ iff M_2 accepts $\langle x, h'(|x|)\rangle$.

The nondeterministic machine M'' discussed above that accepts $L(M)$ with the aid of subpolynomial advice can now be defined. Since M'' will have to simulate M_2, it will require advice. It turns out that the advice M'' requires on input of length n is $h' \circ f(n)$, where

$$f(n) = n + n^{c\epsilon} + bn^{c(\alpha-\beta+\epsilon)}.$$

Note that since f is a polynomial, $|h' \circ f|$ is subpolynomial. 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 $bm^{c\epsilon}$ and $am^{c(\alpha-\beta)}$.

2. Guess configurations $A_0, A_1, \ldots, A_{am^{c(\alpha-\beta)}}$ each of length $bm^{c\epsilon}$ 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 $z = x\#A_1\#A_2\#\ldots\#A_{am^{c(\alpha-\beta)}}$ and simulate M_2 on $w = \langle z, y\rangle$. Note that $|y| = |h'(|z|)|$.

Note that $n \in \Theta(m)$ since $|h' \circ f|$ is sublinear. By Proposition 18, the first step takes linear time. The second step takes time $O(m^{c(\alpha-\beta+\epsilon)})$ which is $O(n^{c(\alpha-\beta+\epsilon)})$. Since M_2 runs in $O(n^{\alpha r+o(1)})$ time and $w \in \Theta(z)$, the third step takes time $O(|z|^{\alpha r+o(1)})$ which is $O(n^{c\alpha r(\alpha-\beta+\epsilon)+o(1)})$. Hence, M'' runs in time $O(n^{c\alpha r(\alpha-\beta+\epsilon)+o(1)})$. By definition of β, the latter is $O(n^{c\alpha r\frac{\alpha+\epsilon}{r+1}+o(1)})$.

Note that M accepts w iff M'' accepts $\langle w, h'(f(|w|))\rangle$. Since $|h' \circ f|$ is subpolynomial, it follows that $L(M)$ is in $\mathsf{NTIME}(n^{c\alpha r\frac{\alpha+\epsilon}{r+1}+o(1)})/n^{o(1)}$. But, x is in L iff M accepts $\langle x, h(|x|)\rangle$. It is easy to see that this implies that L is in $\mathsf{NTIME}(n^{c\alpha r\frac{\alpha+\epsilon}{r+1}+o(1)})/n^{o(1)}$.

Since $L \in \mathsf{coNTIME}(n^c)$ is arbitrary, it follows that

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

Theorem 4 then implies that $\alpha r\frac{\alpha+\epsilon}{r+1} \geq 1$. □

The statement of Theorem 11 was originally weaker than the form given above; Dieter van Melkebeek [17] showed how to alter the original arguments to prove the current version.

5.2 Proof of Theorem 14

The proof of Theorem 14 follows the same basic outline as the proof of Theorem 11. The following technical lemma, whose proof is contained in the appendix, is required in addition to Lemma 17 for the proof of Theorem 14:

Lemma 19. *If* $\mathsf{coNTIME}(n^a) \subseteq \mathsf{NTIME}(n^b)/n^{o(1)}$ *for some constants a and b, then*

$$\mathsf{coNTIME}(n^a)/n^{o(1)} \subseteq \mathsf{NTIME}(n^b)/n^{o(1)}.$$

Proof of Theorem 14. Fix $r \geq 1$ and $\epsilon \leq \alpha/(r+\alpha-1)$ and assume that

$$\mathsf{coNTIME}(n^r) \subseteq \mathsf{NTISP}(n^{\alpha r}, n^{\epsilon r})/n^{o(1)}. \quad (4)$$

This assumption is 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 $n^{c\alpha^2 r\frac{\alpha+\epsilon}{\alpha+r}+o(1)}$ and taking subpolynomial advice. Theorem 4 then gives the desired lower bound for α. The proof builds on the techniques used to prove Theorem 11.

Let

$$c = \max(1/(\alpha-\beta+\epsilon), 1/\epsilon, r),$$

where $\beta = r\frac{\alpha+\epsilon}{\alpha+r}$. Note that $\alpha\beta = r(\alpha-\beta+\epsilon)$. Moreover, since $\epsilon \leq \alpha/(r+\alpha-1)$, it follows that $\beta \geq \epsilon r$.

Let $L \in \mathsf{coNTIME}(n^c)$ be arbitrary. Now, since L is in $\mathsf{coNTIME}(n^c)$, Equation (4) and Lemma 17 imply the existence of a nondeterministic Turing machine M running in time $an^{c\alpha}\log n$ (for some constant a) and space $O(n^{c\epsilon})$, and a function $h : \mathbb{N} \to \Sigma^*$ such that $|h|$ is subpolynomial and $x \in L$ iff M accepts $\langle x, h(|x|)\rangle$. In addition assume, without loss of generality, that M takes exactly $an^{c\alpha}\log n$ steps on each path for all inputs of length n. Moreover, assume without loss of generality, that there exists a constant b such that all configurations of M on input of length n can be described by $bn^{c\epsilon}$ bits.

Note that 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 Equation (4) and Lemma 17, the second "existential" stage in the Σ_3 machine M' can be replaced by a co-nondeterministic computation taking subpolynomial advice, incurring a small time loss, to yield a Σ_2 machine M'' that takes subpolynomial advice in its "universal" stage and accepts $L(M)$. Moreover, by Equation (4) and Lemmas 17 and 19, the "universal" stage in M'' can be replaced by a nondeterministic computation taking subpolynomial advice, again incurring a small time loss, to yield a nondeterministic machine M''' that takes subpolynomial 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'''. So the goal is to construct M''' so that it 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, 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 $bm^{c\epsilon}$ and $m^{c\beta}\log m$. Check that both A and B are valid configurations of M each of length $bm^{c\epsilon}$.

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 18. The second step can be done in time $O(m^{c\epsilon} + m^{c\beta}\log m)$. Note that since $c \geq 1/\epsilon$, it follows that $n = |w| \in \Theta(m^{c\epsilon})$. Hence, M_1 runs in time

$O(n + n^{\beta/\epsilon}\log n)$. Moreover, since $\beta/\epsilon \ge r \ge 1$, the latter is simply $O(n^{\beta/\epsilon}\log n)$. Note now that since $\beta/\epsilon \ge r$, Equation (4) and Lemma 17 imply that there exist a co-nondeterministic Turing machine M_2 running in time $O(n^{\frac{\alpha\beta}{\epsilon}+o(1)})$ and a function h' mapping $\mathbb{N}$ to Σ^* such that $|h'|$ is subpolynomial 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 $A_0, \dots, A_\ell$ of configurations of M, universally guesses an $i < \ell$ and checks that $A_i \vdash_M^* A_{i+1}$ by simulating M_2. Note that M_3 requires advice for the simulation of M_2. As will become apparent below, the required advice on input of length n is $h' \circ f(n)$, where

$$f(n) = \frac{2 + n + bn^{c\epsilon}}{n + (b+1)n^{c\epsilon} + bn^{c(\alpha-\beta+\epsilon)}}.$$

Note that f is polynomially bounded, and thus, $|h' \circ f|$ is subpolynomial. The machine M_3 can now be defined. 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 $bm^{c\epsilon}$ and $m^{c(\alpha-\beta)}$. Check that $\ell = m^{c(\alpha-\beta)}$ and that each of the A_i has length $bm^{c\epsilon}$. Check that $|y| = |h'(2 + m + bm^{c\epsilon})|$.
2. Universally guess an $i < \ell$.
3. Simulate M_2 on input $\langle x\#A_i\#A_{i+1}, y\rangle$.

The first step can be done in linear time with the aid of Proposition 18. The second step also takes linear time. In the third step, M_2 is simulated on input of length $\Theta(m + m^{c\epsilon})$ which is $\Theta(m^{c\epsilon})$ since $c \ge 1/\epsilon$. Since M_2 runs in $O(n^{\frac{\alpha\beta}{\epsilon}+o(1)})$ time, the third step thereby takes $O(m^{c\alpha\beta+o(1)})$ time. Note that since $\alpha\beta = r(\alpha - \beta - \epsilon)$, the latter is $O(m^{cr(\alpha-\beta-\epsilon)+o(1)})$. So M_3 runs in time $O(n + m^{cr(\alpha-\beta-\epsilon)+o(1)})$. Note now that $n \in \Theta(m + m^{c(\alpha-\beta+\epsilon)})$, since $|h'|$ is subpolynomial. In fact, since $c \ge 1/(\alpha - \beta + \epsilon)$, $n \in \Theta(m^{c(\alpha-\beta+\epsilon)})$. Hence, M_3 runs in time $O(n^{r+o(1)})$. Now consider the language L' consisting of those x for which M_3 accepts $\langle x, h'(f(|x|))\rangle$. Then, $L' \in \mathsf{coNTIME}(n^{r+o(1)})/n^{o(1)}$ since $|h' \circ f|$ is subpolynomial. Equation (4) and Lemmas 17 and 19 then imply the existence of a nondeterministic machine M_4 running in time $O(n^{\alpha r+o(1)})$ and a function $h'' : \mathbb{N} \to \Sigma^*$ such that $|h''|$ is subpolynomial and $x \in L'$ iff M_4 accepts $\langle x, h''(|x|)\rangle$.

The nondeterministic machine M''' discussed above that accepts $L(M)$ with the aid of subpolynomial advice can now be defined. It turns out that the advice M''' requires on input of length n is $h'' \circ f'(n)$, where

$$f'(n) = n + n^{c\epsilon} + bn^{c(\alpha-\beta+\epsilon)}.$$

Note that since f is a polynomial, $|h'' \circ f'|$ is subpolynomial. 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 $bm^{c\epsilon}$ and $am^{c(\alpha-\beta)}$.
2. Guess $am^{c(\alpha-\beta)}$ configurations $A_0, \dots, A_{am^{c(\alpha-\beta)}}$ each of length $bm^{c\epsilon}$ 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 $z = x\#A_1\#A_2\#\dots\#A_{am^{c(\alpha-\beta)}}$ and simulate M_4 on $w = \langle z, y\rangle$. Note that $|y| = |h''(|z|)|$.

Note that $n \in \Theta(m)$ since $|h'' \circ f'|$ is sublinear. By Proposition 18, the first step can be done in linear time. The second step takes time $O(m^{c(\alpha-\beta+\epsilon)})$ which is $O(n^{c(\alpha-\beta+\epsilon)})$. 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 $O(n^{c\alpha r(\alpha-\beta+\epsilon)+o(1)})$. Hence, M''' runs in time $O(n^{c\alpha r(\alpha-\beta+\epsilon)+o(1)})$. By definition of β, the latter is $O(n^{c\alpha^2 r\frac{\alpha+\epsilon}{\alpha+r}+o(1)})$.

Note that M accepts w if and only if M''' accepts $\langle w, h''(f'(|w|))\rangle$. Since $|h'' \circ f'|$ is subpolynomial, it follows that $L(M) \in \mathsf{NTIME}(n^{c\alpha^2 r\frac{\alpha+\epsilon}{\alpha+r}+o(1)})/n^{o(1)}$. But, x is in L iff M accepts $\langle x, h(|x|)\rangle$. It is easy to see that this implies that L is in $\mathsf{NTIME}(n^{c\alpha^2 r\frac{\alpha+\epsilon}{\alpha+r}+o(1)})/n^{o(1)}$.

Since $L \in \mathsf{coNTIME}(n^c)$ is arbitrary, it follows that

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

Theorem 4 then implies that $\alpha^2 r\frac{\alpha+\epsilon}{\alpha+r} \ge 1$. Theorem 14 now follows. $\square$

6 Discussion

In [4] 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 tight separations of partially non-uniform and uniform classes, the results in this paper would seem to lend more ammunition to Fortnow's argument.

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 non-uniform, seems quite difficult. Furthermore, substantially strengthening the uniform lower bounds presented in this paper would resolve some long-standing open questions:

Theorem 20. *Suppose there exists an unbounded function $s(n)$ such that,*

$$\mathsf{NTIME}(n^k) \not\subseteq \mathsf{DTISP}(n^{ks(k)}, \log^{O(1)} n),$$

for infinitely many $k \in \mathbb{N}$. Then $\mathsf{SC} \neq \mathsf{NP}$.

Proof. Suppose to the contrary that $\mathsf{SC} = \mathsf{NP}$. Then $\mathbf{SAT}$ is in $\mathsf{DTISP}(n^\ell, \log^{O(1)} n)$ for some integer ℓ. Lemma 3 then implies that

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

for all $j > \ell$, contradicting the hypotheses. $\square$

Further strengthening the hypotheses in Theorem 20 yields an even more impressive separation:

Theorem 21. *Suppose there exists an unbounded function $s(n)$ such that,*

$$\mathsf{NTIME}(n^k) \not\subseteq \mathsf{DTIME}(n^{ks(k)})$$

for infinitely many $k \in \mathbb{N}$. Then $\mathsf{P} \neq \mathsf{NP}$.

However, it seems that substantially stronger techniques are needed to even get the separations required in the hypotheses of Theorem 20.

Showing the existence of a language in $\mathsf{NTIME}(n^k) - (\mathsf{NTIME}(o(n^k))/n)$ seems achievable with the present techniques. On the other hand, increasing the amount of non-uniformity in Theorem 4, and thereby possibly the amount of non-uniformity in all the non-uniform results presented in this paper, seems quite difficult. Indeed, increasing the amount of uniformity to $n^{1+o(1)}$ in Theorem 4 would imply the existence of a language in NP without linear (completely non-uniform) circuits.

Acknowledgements

I would like to thank Valentine Kabanets, Steve Myers, Charlie Rackoff and, in particular, Steve Cook for many helpful discussions and comments.

I would also like to 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] R. V. Book, S. A. Greibach, and B. Wegbreit. Time- and tape-bounded Turing acceptors and AFLs. *Journal of Computer and System Sciences*, 4(6):606–621, 1970.

[2] S. A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the 3rd ACM Symposium on the Theory of Computating*, pages 151–158, ACM, New York, 1971.

[3] S. A. Cook. Short propositional formulas represent nondeterministic computations. *Information Processing Letters*, 26(5):269–270, Jan. 1988.

[4] L. Fortnow. Nondeterministic polynomial time versus nondeterministic logarithmic space: Time-space tradeoffs for satisfiability. In *Proceedings, Twelfth Annual IEEE Conference on Computational Complexity*, pages 52–60, Ulm, Germany, 24–27 June 1997. IEEE Computer Society Press.

[5] L. Fortnow and D. van Melkebeek. Time-space tradeoffs for nondeterministic computation. Submitted, 2000.

[6] F. C. Hennie and R. E. Stearns. Two-tape simulations of multitape Turing machines. *Journal of the Association for Computing Machinery*, 13(4):533–546, Oct. 1966.

[7] J. E. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, Reading, Massachusetts, 1979.

[8] R. Kannan. Towards separating nondeterminism from determinism. *Mathematical Systems Theory*, 17(1):29–45, Apr. 1984.

[9] 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*, pages 302–309, Los Angeles, California, 28–30 Apr. 1980.

[10] R. J. Lipton and A. Viglas. On the complexity of SAT. In *40th Annual Symposium on Foundations of Computer Science*, pages 459–464, New York City, New York, 17–19 Oct. 1999. IEEE.

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

[12] C. Papadimitriou. *Computational Complexity*. Addison-Wesley, Reading, Massachusetts, 1994.

[13] 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*, pages 429–438, Tucson, Arizona, 7–9 Nov. 1983. IEEE.

[14] N. Pippenger and M. J. Fischer. Relations among complexity measures. *Journal of the Association for Computing Machinery*, 26(2):361–381, Apr. 1979.

[15] C. Rackoff. Personal communication, 1999.

[16] J. M. Robson. An $O(T \log T)$ reduction from RAM computations to satisfiability. *Theoretical Computer Science*, 82(1):141–149, May 1991.

[17] D. van Melkebeek. Personal communication, 2000.

[18] A. Viglas. Personal communication, 1999.

A Proofs of Lemmas 17 and 19

Proof of Lemma 17. Fix r, q and α and suppose that

$$\mathsf{coNTIME}(n^r) \subseteq \mathsf{NTISP}(n^{\alpha r}, n^{\beta r})/n^{o(1)}.$$

Let $L \subseteq \Sigma^*$ be in $\mathsf{coNTIME}(n^q)$. Consider the language

$$L' = \{x\#^{f(|x|)-|x|} : x \in L\},$$

where $\# \notin \Sigma$ and $f(n) = n^{q/r}$. It is clear that L' is in $\mathsf{coNTIME}(n^r)$ and hence in $\mathsf{NTISP}(n^{\alpha r}, n^{\beta r})/n^{o(1)}$. In particular, there exist a Turing machine M with j work tapes that runs in $n^{\alpha r}$ time and $n^{\beta r}$ space and a function h mapping $\mathbb{N}$ to Σ^* such that $g(n) = |h(n)|$ is constructible and subpolynomial, and $x \in L$ iff M accepts $\langle x, h(|x|)\rangle$.

Note that $|h \circ f| = g \circ f$ is subpolynomial. So to prove the lemma it suffices to exhibit a Turing machine M' running in time $O(n^{\alpha q} \log n)$ and space $O(n^{\beta q})$ such that x is in L iff $\langle x, h(f(|x|))\rangle$ is accepted by M'. To that end, consider a Turing machine M' with $j + 2$ work tapes 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 the first tape. Check that $|y| = g(f(|x|))$. Since $g \circ f$ is sublinear, the latter implies that $|x| \in \Theta(n)$.

2. Simulate M on input $z = \langle x\#^{f(|x|)-|x|}, y\rangle$ on the last j tapes and accept iff M accepts. Note that $|z| \in \Theta(f(|x|)) = \Theta(f(n))$. Machine M' does not actually write down the input z. Instead, it uses its second tape to write down in binary the position of M's input head. The second tape is updated accordingly after each simulated move of M. In particular, the second tape initially has the number 0 written on it. Whenever M's input head must be moved left (right) during the simulation, the number on the second tape is decremented (incremented) by 1. If the value i on the second tape is greater than n, then the symbol under M's (simulated) input head must be $\#$; otherwise, the symbol under M's (simulated) input head is the ith symbol on M''s input tape.

Since g is time-constructible, the first step can be done in linear time and logarithmic space with the aid of Proposition 18. Note that it takes $O(\log n)$ for M' to simulate a step of M since M' has to update the index on the second tape after each simulated step. Hence, the simulation of M by M' takes 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 $O(n^{\beta q})$.

So M' uses $O(n^{\alpha q} \log n)$ time and $O(n^{\beta q})$ space. Moreover, as desired, x is in L iff M' accepts $\langle x, h(f(|x|))\rangle$. The lemma now follows. $\square$

Proof of Lemma 19. Assume the hypotheses and suppose that L is in $\mathsf{coNTIME}(n^a)/n^{o(1)}$. Then there exist a nondeterministic machine M running in time n^a and a function h mapping $\mathbb{N}$ to Σ^* such that $|h|$ is constructible and subpolynomial, and $x \in L$ iff M accepts $\langle x, h(|x|)\rangle$. Then by the hypotheses there exist a deterministic Turing machine M' running in time n^b and a function h' mapping $\mathbb{N}$ to Σ^* such that $|h'|$ is constructible and subpolynomial, and M accepts y iff M' accepts $\langle y, h'(|y|)\rangle$.

Define $g(n)$ to be the length of $\langle x, y\rangle$ when $|x| = n$ and $h(|n|)$. Since $|\langle x, y\rangle|$ only depends on $|x|$ and $|y|$, g is well-defined. Now consider the function

$$h''(n) = \langle h(n), h'(g(n))\rangle.$$

Note that since $|\langle x, y\rangle| \leq 4(|x| + |y|)$, and $|h|$ and $|h'|$ are subpolynomial, it follows that $|h''(n)|$ is also subpolynomial.

Now consider a deterministic machine M'' that on input $w = \langle x, h''(|x|)\rangle$, $|w| = n$, computes as follows:

1. Check that the input has the right form and write down $h(|x|)$ and $h'(g(|x|))$.

2. Simulate M' on input $z = \langle y, h'(|y|)\rangle$ where $y = \langle x, h(|x|)\rangle$. Note that $h'(|y|) = h'(g(|x|))$.

The first step takes linear time. The second step takes time $|z|^b$. Now, since $|h''|$ is sublinear, it follows that $|x| \in \Theta(n)$. Moreover, since $|h|$ and $|h'|$ are subpolynomial, $|z| \in \Theta(|x|)$. Hence $|z| \in \Theta(n)$. So the second step takes time $O(n^b)$. Thus M'' runs in time n^b.

Now, $x \in L$ iff M accepts $\langle x, h(|x|)\rangle$ iff M' accepts $\langle y, h'(|y|)\rangle$, where $y = \langle x, h(|x|)\rangle$, iff M'' accepts $\langle x, h''(|x|)\rangle$. Hence, $L \in \mathsf{NTIME}(n^b)/n^{o(1)}$. The lemma now follows since $L \in \mathsf{coNTIME}(n^a)/n^{o(1)}$ is arbitrary. $\square$