A Time Lower Bound for Satisfiability ⋆

Dieter van Melkebeek [a,*,1]

[a] *Department of Computer Sciences, Univesity of Wisconsin, Madison, WI 53706, USA.*

Ran Raz [b]

[b] *Faculty of Mathematics and Computer Science, Weizmann Institute of Science, Rehovot 76100, Israel.*

Abstract

We show that a deterministic Turing machine with one d-dimensional work tape and random access to the input cannot solve satisfiability in time n^a for $a < \sqrt{(d+2)/(d+1)}$. For conondeterministic machines, we obtain a similar lower bound for any a such that $a^3 < 1 + a/(d+1)$. The same bounds apply to almost all natural NP-complete problems known.

Key words: Computational Complexity, Lower bounds, Satisfiability
1991 MSC: 68Q10, 68Q15, 68Q25

⋆ A preliminary version of this paper appeared in the Proceedings of the 31st International Colloquium on Automata, Languages and Programming.
* Corresponding author.
Email addresses: `dieter@cs.wisc.edu` (Dieter van Melkebeek), `ran.raz@weizmann.ac.il` (Ran Raz).
URLs: `www.cs.wisc.edu/~dieter` (Dieter van Melkebeek), `www.wisdom.weizmann.ac.il/~ranraz` (Ran Raz).
[1] Partially supported by NSF Career award CCR-0133693.

1 Introduction

Proving time lower bounds for natural problems remains the most difficult challenge in computational complexity. We know exponential lower bounds on severely restricted models of computation (e.g., for parity on constant depth circuits) and polynomial lower bounds on somewhat restricted models (e.g., for palindromes on single tape Turing machines) but no nontrivial lower bounds on general random-access machines. In this paper, we exploit the recent time-space lower bounds for satisfiability on general random-access machines to establish new lower bounds of the second type, namely a time lower bound for satisfiability on Turing machines with one multidimensional work tape and random access to the input.

1.1 Lower Bounds for Satisfiability

Satisfiability constitutes the seminal NP-complete problem and is of major practical importance. While we expect the problem to take time $2^{\Omega(n)}$ in the worst case, the sad state of affairs is that we cannot even rule out the existence of a linear-time algorithm on a random-access Turing machine.

We do have nontrivial lower bounds on the running time of random-access Turing machines that solve satisfiability in sublinear space. We have seen considerable progress on such time-space lower bounds in recent years [9,4,10,5]. The state-of-the-art is a time lower bound of essentially n^{ϕ} for algorithms using subpolynomial space, where ϕ denotes the golden ratio, about 1.618. More precisely, the following holds:

Theorem 1 (Fortnow-Van Melkebeek [5]) *Let $\phi \doteq (\sqrt{5}+1)/2$ denote the golden ratio. For any constant $a < \phi$ there exists a positive constant b such that satisfiability cannot be solved on a deterministic random-access Turing machine in time n^a and space n^b.*

A nice feature of Theorem 1 is its model independence – the proof works for any reasonable model of computation. However, the theorem does not yield any lower bounds for algorithms that use linear space, e.g., algorithms that explicitly store an assignment to the given formula.

An almost quadratic time lower bound for satisfiability on single tape Turing machines immediately follows from the quadratic lower bound for palindromes in that model because of the standard efficient translation of any problem in NP to satisfiability. This result does not rely on the inherent difficulty of satisfiability, though. It rather exploits an artifact of the single tape Turing machine model – that the machine has to waste a lot of time in moving its tape head between both ends of the tape in order to retrieve information about the input. As soon as we include a work tape separate from the input tape, palindromes can be decided in linear time.

1.2 Our Results

We consider models of computation whose power lies between single tape Turing machines and random-access Turing machines, and establish time lower bounds of the form n^a where a is a constant larger than 1. Our proofs rely on the fact that satisfiability captures nondeterministic computation.

The first model we consider is that of a Turing machine with two tapes, namely an input tape and one work tape. The model is known as the single tape off-line Turing machine, and constitutes the strongest model with two-way access to the input on which superlinear time lower bounds for natural decision problems were established. Maass et al. [11] proved a lower bound of $\Omega(n \log n)$ for a problem in P, and Kannan [8] sketched a lower bound of $n^{1.104}$ for satisfiability. We improve Kannan's lower bound to n^a for any constant $a < \sqrt{3/2} \approx 1.224$. In fact, our result also holds if we allow random access to the input.

We generalize our lower bound to the case of Turing machines with a d-dimensional work tape.

Theorem 2 (Main Result) *For any positive integer d and any constant $a < \sqrt{(d+2)/(d+1)}$, satisfiability cannot be solved in time n^a on a deterministic Turing machine with a d-dimensional work tape and random access to the input.*

Dietzfelbinger and Hühne [3] proved a polynomial lower bound in this model but with the additional restriction that the input tape is one-way. Theorem 2 provides the first superlinear time lower bound for Turing machines with a planar or higher dimensional work tape and random-access to the input.

Our approach also applies to conondeterministic algorithms for satisfiability, or equivalently, to nondeterministic algorithms for tautologies.

Theorem 3 *For any positive integer d and any constant a such that $a^3 < 1 + a/(d+1)$, satisfiability cannot be solved in time n^a on a conondeterministic Turing machine with a d-dimensional work tape and random access to the input.*

The bound in Theorem 3 is somewhat weaker than the one in Theorem 2. Let $g(d)$ denote the solution $a > 1$ of the equation $a^3 = 1 + a/(d+1)$. The function $g(d)$ lies somewhere between the deterministic bound $f(d) \doteq \sqrt{(d+2)/(d+1)}$

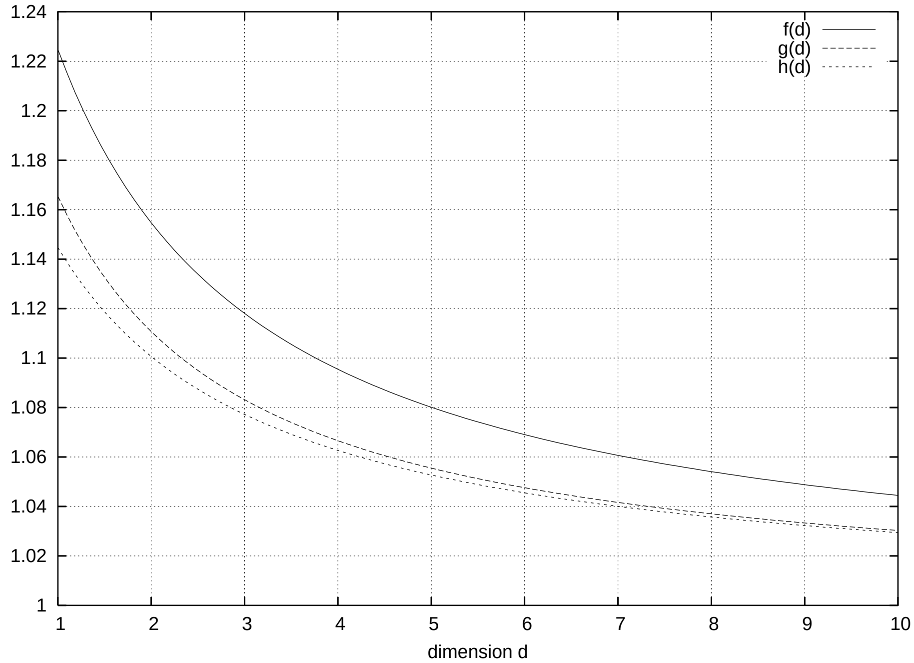

Fig. 1. Plot of the exponent a as a function of the single work tape's dimension d in the time lower bounds of the form n^a for satisfiability: the deterministic bound $f(d)$, the conondeterministic bound $g(d)$, and the lower bound $h(d) = (f(d))^{2/3}$ for $g(d)$.

and $h(d) \doteq (f(d))^{2/3} = \sqrt[3]{(d+2)/(d+1)}$. See Figure 1.2 for a plot of these functions.

Time lower bounds for satisfiability immediately imply time lower bounds for problems to which satisfiability efficiently reduces. Almost all known natural NP-complete problems translate to satisfiability in quasilinear ($n \cdot \text{poly} \log n$) time such that each bit of the translation can be computed in polylogarithmic time on a random-access Turing machine. As a corollary to Theorems 2 and 3, we can extend our lower bounds to all such problems.

Corollary 1 *The lower bounds of Theorems 2 and 3 apply to any problem to which satisfiability Karp reduces in time $n^{1+o(1)}$ on a random-access Turing machine such that each bit of the reduction can be computed in time $n^{o(1)}$.*

1.3 Our Approach

Our starting point is the recent time-space lower bounds for satisfiability on random-access machines (see [14] for a survey). The high-level structure of these arguments is that of a proof by indirect diagonalization. We start from the assumption that satisfiability has a deterministic algorithm that runs in time t and space s. Since satisfiability captures nondeterministic (quasi-)linear time in a very strong sense, we can roughly view our assumption as the inclusion

$$\mathsf{NTIME}(n) \subseteq \mathsf{DTISP}(t, s), \tag{1}$$

where $\mathsf{DTISP}(t, s)$ denotes the class of problems that can be solved deterministically in time t and space s simultaneously. Then we use (1) to derive more and more unlikely inclusions of complexity classes, up to the point where we reach a contradiction with a diagonalization result.

A crucial step in the proof of Theorem 1 is an inclusion of the form

$$\mathsf{DTISP}(T, S) \subseteq \mathsf{NTIME}(f(T, S)), \tag{2}$$

where $f(T, S) \ll T$, which actually follows from a weaker hypothesis than (1), namely

$$\mathsf{NTIME}(n) \subseteq \mathsf{DTIME}(t). \tag{3}$$

Inclusion (2) describes a speedup of deterministic space bounded computations on nondeterministic machines and is proved by a combination of the following two arguments.

- We can speed up $\mathsf{DTISP}(T, S)$ computations on an alternating machine by breaking up the computation tableau into b blocks, guessing the configurations at the $b-1$ common boundaries of the blocks, and universally verifying the computation on each of the blocks of size T/b. This yields the inclusion

  $$\mathsf{DTISP}(T, S) \subseteq \Sigma_2\mathsf{TIME}(b \cdot S + T/b).$$

  Applying this idea k times recursively with block numbers $b_1, b_2, \ldots, b_k$, respectively, and exploiting the closure under complementation of deterministic classes to save about half of the alternations [5], we get

  $$\mathsf{DTISP}(T, S) \subseteq \Sigma_{k+1}\mathsf{TIME}((\sum_j b_j) \cdot S + T/(\prod_j b_j)). \tag{4}$$

- We can eliminate alternations using the hypothesis (3). If $t(n)$ is of the form n^a for some constant a, (3) allows us to eliminate one alternation from an alternating computation at the cost of raising the running time to the power a. Eliminating all k alternations of the right-hand side of (4) from back to front yields a nondeterministic simulation running in time $f(T, S)$, where the actual form of $f(T, S)$ depends on a and the choice of $b_1, b_2, \ldots, b_k$.

The proof of Theorem 1 then proceeds as follows. For any smooth bound $\tau(n) \geq n$, the hypothesis (1) implies an inclusion of the form $\mathsf{NTIME}(\tau) \subseteq$

$\mathsf{DTISP}(T, S)$. Combining with (2) leads to the conclusion $\mathsf{NTIME}(\tau) \subseteq \mathsf{NTIME}(f(T, S))$, which contradicts the nondeterministic time hierarchy theorem as long as $f(T, S) = o(\tau)$. The rest of the proof of Theorem 1 involves selecting optimal values for the number of alternations k and the block numbers $b_1, b_2, \ldots, b_k$ so as to minimize the function $f(T, S)$.

Now, suppose we try a similar strategy to obtain a *time* lower bound instead of a time-space lower bound. Thus, our aim is to derive a contradiction from the hypothesis $\mathsf{NTIME}(n) \subseteq \mathsf{DTIME}(t)$ where t is of the form $t(n) = n^a$ for as large a constant a as possible. Note that we can still exploit the speedup of space bounded computations by nondeterminism given by (2) since that step only used the hypothesis (3). The problem is to obtain a deterministic simulation of $\mathsf{NTIME}(\tau)$ that runs in small space. Such a simulation immediately follows from the stronger hypothesis (1) but we do not know how to derive it from the weaker hypothesis (3) when the underlying model of computation allows random memory access. In case of sequential memory access, however, we can use the notion of crossing sequences [6] to break up the computation into pieces that each run in small space, and then apply (2).

Consider a deterministic computation that takes t steps on a Turing machine with a single work tape and random access to the input. We can simulate such a computation on an alternating random-access machine as follows: Break up the tape into b blocks of size t/b each. Guess the crossing sequences at all the block boundaries. By choosing an appropriate offset for the blocks, we can argue that the total number of crossings we need to guess is no more than b. Then switch to a universal mode and verify the computation on each of the b blocks given the crossing sequences for that block. The verification for a given block can be performed in time $T = t$ and space $S = t/b$. This gives us the time-space bounded computation that is critical for the argument of Theorem 1. We can speed up (the complement of) that computation as in (4) and obtain a simulation that essentially lives in

$$\Sigma_{k+2}\mathsf{TIME}(b + (\sum_{j\geq 1} b_j) \cdot S + T/(\prod_{j\geq 1} b_j)). \tag{5}$$

Now, suppose there exists a Turing machine with a single work tape and random access to the input that solves satisfiability in time t. Since random-access machines can efficiently simulate sequential machines, we have that, roughly, $\mathsf{NTIME}(n) \subseteq \mathsf{DTIME}(t)$, so we can eliminate alternations at the cost of a small increase in running time as before. Using a similar argument as in the proof of Theorem 1, we obtain a contradiction to the nondeterministic time hierarchy theorem for small t. It turns out that $k = 1$ leads to the strongest results for this approach – we can rule out running times $t(n)$ up to n^a for $a < \sqrt[3]{3/2}$.

We can do better by exploiting the following slack in our argument. We modeled the verification of any given block as a computation that takes time $T = t$ and uses space $S = t/b$. We cannot improve the upper bound $T = t$ for all blocks since it is possible for the computation to spend all its time in one particular block. On average, though, the time the computation spends on a block will be much less. We can benefit as follows from the fact that the total time spent on all blocks together is at most t.

Let t_i denote the time spent on block i. At the second existential level of (5), for a given block i, we guess a configuration after each t/b_1 steps the computation spends on block i. Thus, we really only need to guess $b_1 t_i/t$ configurations for block i at that level. The total number of configurations we guess at the second existential level is therefore bounded by $\sum_i b_1 t_i/t = b_1$. We can as well guess all these b_1 configurations at the first existential level. This saves us one alternation, leading to a simulation that lives in

$$\Sigma_{k+1}\mathsf{TIME}(b + (\sum_{j\geq 1} b_j) \cdot S + T/(\prod_{j\geq 1} b_j)). \tag{6}$$

Using this improvement, we manage to rule out running times $t(n)$ up to n^a for $a < \sqrt{3/2}$.

A more direct way of obtaining a simulation of type (6) is the following. The above verification process for all blocks combined can be executed on a random access machine in roughly time $O(t)$ and space $O(S)$. Thus, we can transform a deterministic time t computation on a Turing machine with one work tape and random access to the input into (i) nondeterministically guessing an offset and at most b crossings, followed by (ii) a deterministic computation that runs in time $O(t)$ and space $O(S)$ on a random access machine. Applying (4) to (ii) results in a simulation of type (6). Similar strategies to reduce the space complexity for one-tape off-line Turing machine computations have been considered in the past but either incurred an additional cost in running time due to the use of one-tape off-line Turing machines for the simulation [7,12], or else involved a nontrivial number of alternations [8].

Our arguments carry over to Turing machines with a d-dimensional work tape and random access to the input, as well as to conondeterministic machines.

1.4 Organization

In Section 2, we describe the various machine models we consider in this paper, and provide the required technical details of the known time-space lower bounds for satisfiability. Section 3 contains the derivation of our main result for Turing machines with a one-dimensional work tape and random access to the input. In Section 4, we extend that result to Turing machines with one d-dimensional work tape for arbitrary positive integers d, to conondeterministic Turing machines, and to NP-complete problems other than satisfiability.

2 Preliminaries

2.1 Machine Models

We use two different machine models – one with sequential memory access and one with random memory access. Both have random read access to the input.

Our main result holds for a sequential memory access model with one d-dimensional work tape for some positive integer d. The work tape has one tape head. In each computation step, the memory cell under the tape head can be accessed (read and/or written) and the tape head can be moved to a neighboring memory cell.

Our proofs also make use of machines with random memory access. We model random access using an auxiliary index tape. An index tape acts as a one-dimensional one-way write-only tape. In any given computation step, the machine can decide to access the cell indexed by the contents of the auxiliary index tape, after which the auxiliary index tape is automatically reset.

The random memory access model can simulate the sequential memory access model with a logarithmic overhead in time.

All notation for complexity classes, e.g., $\mathsf{NTIME}(t)$, refers to the random memory access model. We omit explicit constructibility conditions and other smoothness requirements on time and space bounds. Eventually, we only need to consider polynomial bounds, which meet all conditions needed.

2.2 Time-Space Lower Bounds for Satisfiability

We use a number of ingredients from the known time-space lower bounds for satisfiability. First, a reduction capturing the very close relationship between satisfiability and nondeterministic computation.

Lemma 1 (Cook [2]) *There exists a constant c such that for every language $L \in \mathsf{NTIME}(\tau)$ where $\tau(n) \geq n$, there exists a reduction from L to* *satisfiability that maps an input x of length n to a formula ϕ_x of length $N \leq \tau(n) \cdot (\log \tau(n))^c$. Moreover, given x and an index i, the ith bit of ϕ_x can be computed in time $(\log \tau(n))^c$.*

Second, we exploit the following crucial ingredient, which quantifies a speedup of deterministic time-space bounded computations on nondeterministic machines that follows if we can simulate nondeterminism very efficiently on deterministic machines.

Lemma 2 (Fortnow-van Melkebeek [5]) *Suppose that*

$$\mathsf{NTIME}(n) \subseteq \mathsf{DTIME}(n^a)$$

for some constant $a \geq 1$. Then for any integer $k \geq 0$ and functions $T(n)$ and $S(n)$,

$$\mathsf{DTISP}(T, S) \subseteq \mathsf{NTIME}((T \cdot S^k)^{c_k} + (n + S)^{a^k}), \tag{7}$$

where $c_0 = 1$ and $c_{k+1} = ac_k/(1 + c_k)$.

Finally, we also use the nondeterministic time hierarchy theorem.

Lemma 3 (Seiferas-Fischer-Meyer [13]) *Let $\tau_1(n)$ and $\tau_2(n)$ be time bounds. If $\tau_1(n + 1) \in o(\tau_2(n))$ then*

$$\mathsf{NTIME}(\tau_2) \not\subseteq \mathsf{NTIME}(\tau_1).$$

In case $\tau_1(n) = n^{e_1}$ and $\tau_2(n) = n^{e_2}$ where e_1 and e_2 are positive constants, Lemma 3 implies that nondeterministic machines can do strictly more in time τ_2 than in time τ_1 if $e_2 > e_1$ [1].

3 Result for One-Dimensional Tapes

In this section, we derive our time lower bound for satisfiability on deterministic machines with a one-dimensional work tape and random access to the input. We refer to Section 1.3 of the introduction for the intuition behind the derivation and follow the direct approach discussed at the end of that section.

The proof goes by contradiction. We start from the hypothesis that satisfiability can be solved by a machine M with one work tape and random access to the input in time n^a for some constant $a \geq 1$. We then argue that for $a < \sqrt{3/2}$, this hypothesis leads to a contradiction with the nondeterministic time hierarchy theorem.

Let L be a language in $\mathsf{NTIME}(\tau)$ for some smooth function $\tau(n) \geq n$ which we will specify later. Let x be an input of length n and ϕ_x the Boolean formula of length $N \leq \tau(n) \cdot (\log \tau(n))^c$ that captures the membership of x to L, as given by Lemma 1. We decide the membership of x to L by simulating M on input ϕ_x on a random-access machine. Since each bit of ϕ_x can be computed on the fly in time poly $\log \tau$, the running time of the simulation is at most a factor poly $\log \tau$ times the running time of simulating M on ϕ_x when ϕ_x is given as input.

Consider the computation of M on input ϕ_x. Since M runs in time at most $t \doteq N^a$, M cannot access any memory cells outside the initial segment of length t. We break up this initial segment into $b + 1$ consecutive blocks of roughly equal size S, and number the blocks 0 through b. More precisely, all blocks except possibly blocks 0 and b contain exactly S cells, and blocks 0 and b contain no more than S cells. See Figure 2, where f denotes the number of cells in block 0.

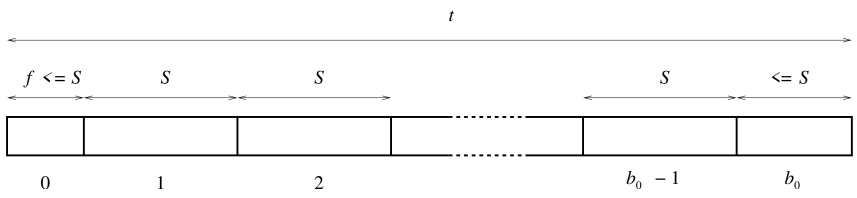

Fig. 2. Breaking up the work tape

Note that f and S fully determine the blocks. The value of b is essentially equal to t/S; more precisely, $t/S - 1 \leq b < t/S + 1$. The parameter S will be set later. We now determine a value for f.

For a given partition into blocks, the computation of M induces a crossing sequence at the boundary of any two consecutive blocks. The crossing sequence at a given boundary consists of a collection of records, one for each time the tape head crosses that boundary. The record corresponding to a particular crossing contains the time step of the crossing, its location on the tape, and the internal state of M as well as the configuration of the index tape for the input at the time of the crossing. Note that each crossing record involves $O(\log t)$ bits of information.

Let X_f denote the list of all crossings over all the block boundaries for a given value of f (and S). By choosing the offset f appropriately, we can ensure that X_f contains no more than $b+1$ crossings. This is because the sets X_f, $1 \leq f \leq S$, form a partition of the collection of all crossings (over all boundaries between consecutive memory cells) during the computation. Since there can be at most one crossing per time step, the total number of crossings is no more than t. By averaging, there exists an offset f, $1 \leq f \leq S$, such that X_f contains no more than $t/S \leq b + 1$ crossings.

Once we know f and X_f, we can break up the computation of M into $b+1$ independent parts, namely one for each block. We can check the correctness of X_f by verifying, for each block i, the consistency of the left end and the right end crossings. The verification process for block i involves the following:

- If $i > 0$ and there is no crossing record for the left boundary then accept if there is no crossing record for the right boundary either and reject otherwise.
- Initialize block i as empty.
- Start from the time, internal state of M, and contents of the index tape as specified in the first crossing record for the left boundary (which for $i = 0$ default to time 0, M's initial state, and an empty index tape, respectively). Simulate M up to the point where M either halts or leaves block i. In case M halts, accept if M accepts and reject otherwise. If not, verify that the current time, internal state of M, and contents of the index tape are as specified in the next crossing record for the boundary being crossed (which defaults to the first crossing record in case of crossing the right boundary). Reject if there is no such record or if there is no consistency.
- If there is no next crossing record for the same boundary that was just crossed, accept if there are no further crossing records for the other boundary either and reject otherwise. If there is a next crossing record for the same boundary, then continue the process in the previous step starting from that record.

If we require the crossing sequences in X_f to be ordered by boundary from left to right, and within a given boundary by increasing time stamp, the verification process for a given block i can be executed in time $O(t_i \cdot \log^{O(1)} t)$ and space $O(S + \log t)$ on a random access Turing machine, where t_i denotes the time M spends on block i. Thus, given f and X_f, the overall verification process runs in time $T = O(t \log^{O(1)} t)$ and space $O(S + \log t)$ on a random access Turing machine. By incorporating some clocking, we can make sure the same holds for every f and list of crossings X.

Let us denote by $\mathrm{Accept}_M(x, f, X)$ the predicate that the tests for all blocks i are passed on input x, offset f, and list of crossings X. By the above, we can decide membership of x to L by evaluating the following condition:

$$(\exists \text{ offset } 1 \leq f \leq S)(\exists \text{ set } X \text{ of at most } b \text{ crossings}) \, \mathrm{Accept}_M(x, f, X). \quad (8)$$

We now analyze how efficiently we can evaluate (8) on a nondeterministic random access machine based on our hypothesis. In order to simplify the expressions, we neglect multiplicative factors of the form $\log^{O(1)} t$.

Since $\mathrm{Accept}_M(x, f, X)$ involves a $\mathsf{DTISP}(T, S)$ computation on an input of length $O(n + b)$, Lemma 2 allows us to transform (8) into a nondeterministic computation running in time

$$O(b + (T \cdot S^k)^{c_k} + (n + b + S)^{a^k}) \tag{9}$$

for any integer $k \geq 0$. We obtain a contradiction with the nondeterministic time hierarchy theorem provided (9) is $o(\tau(n-1)/\text{poly}\log\tau(n))$. Our goal now is to select the parameters in such a way that we obtain that contradiction for values of a as large as possible.

Recall that $T = t$, $S = t/b$, and $t = \tau^a$. Setting $b = t^\beta$ for some constant $\beta \in (0, 1)$ and letting $\tau(n) = n^e$ for a sufficiently large constant e, we obtain the contradiction we seek as long as

$$a \cdot \max((1 + k(1 - \beta))c_k, \beta a^k, (1 - \beta)a^k) < 1. \tag{10}$$

The first interesting value of k is $k = 1$. For $k = 1$, requirement (10) simplifies to

$$a^2 \cdot \max(1 - \beta/2, \beta) < 1,$$

for which the optimal choice of $\beta = 2/3$ leads to the bound $a < \sqrt{3/2}$.

Further calculations show that values of $k \geq 2$ do not lead to better bounds on a. By dropping the first component of the maximum expression in (10), the optimal setting of $\beta = 1/2$ leads to the necessary condition that $a^{k+1} < 2$, which is stricter than $a < \sqrt{3/2}$ for $k > 2$. For $k = 2$, dropping the second term of the maximum expression in (10) and setting β optimally to $\beta = 1 - 1/a$ results in the necessary condition $a^2 < 1$. Thus, the result claimed in the statement of Theorem 2 for $d = 1$ is the best we can get using our approach.

4 Extensions

In this section, we extend the result from the previous section for machines with a one-dimensional work tape to machines with a d-dimensional work tape for arbitrary positive integers d. We also derive a similar lower bound for conondeterministic machines, and argue that all our bounds apply to NP-complete problems other than satisfiability.

4.1 Multi-Dimensional Tapes

We follow the proof outline of Section 3. We assume that satisfiability can be solved on a machine M with one d-dimensional tape and random access to the input in time n^a for some constant $a \geq 1$, and argue that this assumption leads to a contradiction with the nondeterministic time hierarchy theorem for $a < \sqrt{(d+2)/(d+1)}$.

We now break up each tape dimension into b parts. The number of blocks becomes b^d; the number of possible offsets f as well as the space occupied by a single block becomes $S = (t/b)^d$.

There still exists a choice for the offset f such that the set of crossings X_f is of size at most b. The averaging argument can be modified as follows. Any given crossing that occurs during the computation of M can appear in up to $(t/b)^{d-1}$ of the sets X_f. This is because the crossing fixes the component of f in the dimension of the crossing but leaves the remaining $d-1$ components of f free. Thus, we get the inequality $\sum_f |X_f| \leq t \cdot (t/b)^{d-1}$. Since there are $(t/b)^d$ possible offsets, this implies that there exists at least one offset f for which $|X_f| \leq b$.

With these modified parameters, (8) still holds, as well as the bound (9). Using the same settings as before, condition (10) generalizes to

$$a \cdot \max((1 + kd(1-\beta))c_k, \beta a^k, d(1-\beta)a^k) < 1. \tag{11}$$

For $k = 1$, the first interesting value for k, (11) becomes

$$a^2 \cdot \max((1 + d(1-\beta))/2, \beta, d(1-\beta)) < 1,$$

for which the optimal choice of $\beta = (d+1)/(d+2)$ leads to the bound $a < \sqrt{(d+2)/(d+1)}$.

Again, $k = 1$ turns out to give the best results. This can be seen as follows. Dropping the first term in the maximum expression in (11) and setting β optimally to $\beta = d/(d+1)$ leads to the necessary condition that $a^{k+1}d/(d+1) < 1$, which is more stringent than $a < \sqrt{(d+2)/(d+1)}$ for $k > 1$ and $d \geq 2$.

Note that a sufficient strengthening of Theorem 2 for general dimension d would imply a time lower bound for satisfiability on multi-tape Turing machines. This follows from the well-known simulation of a k-tape Turing machine running in time t on a Turing machine with one d-dimensional work tape in time $O(t\sqrt[d]{t})$ [15].

4.2 Conondeterministic Machines

The argument used in the proof of Theorem 2 can be modified for conondeterministic instead of deterministic machines.

There are two key modifications. The first one is the use of the following diagonalization result instead of the nondeterministic time hierarchy theorem given in Lemma 3.

Lemma 4 *Let $\tau(n)$ be a time bound.*

$$\mathsf{coNTIME}(\tau) \not\subseteq \mathsf{NTIME}(o(\tau)).$$

Thus, assuming there exists a conondeterministic Turing machine with one d-dimensional work tape and random access to the input that solves satisfiability in time $t(n) = n^a$, we aim for a contradiction to Lemma 4 by showing that an arbitrary language $L \in \mathsf{coNTIME}(\tau)$ can be simulated on a nondeterministic machine in time $o(\tau)$.

We can decide L by evaluating the predicate (8), in which the matrix Accept_M now involves a $\mathsf{NTISP}(T, S)$ computation (instead of $\mathsf{DTISP}(T, S)$) on an input of length $O(n + b \log t)$. The second key modification is that we apply the following lemma instead of Lemma 2 to speed up the computation of Accept_M.

Lemma 5 (Fortnow-van Melkebeek [5]) *Suppose that*

$$\mathsf{NTIME}(n) \subseteq \mathsf{coNTIME}(n^a)$$

for some constant $a \geq 1$. Then for any integer $k \geq 0$ and functions $T(n)$ and $S(n)$,

$$\mathsf{NTISP}(T, S) \subseteq \mathsf{NTIME}((T \cdot S^k)^{g_k} + (n + S)^{a^{2k}}),$$

where $g_0 = 1$ and $g_{k+1} = a^2 g_k / (1 + a g_k)$.

After applying Lemma 5, (8) turns into a nondeterministic algorithm for deciding L that runs in time

$$b + (T \cdot S^k)^{g_k} + (n + b + S)^{a^{2k}} \tag{12}$$

times a term of the form $\mathrm{poly} \log t$. We obtain a contradiction with Lemma 4 provided (12) is $o(\tau / \mathrm{poly} \log \tau)$. Setting the parameters as before, we get this contradiction as long as there exists an integer $k \geq 0$ and a constant $\beta \in (0, 1)$ such that

$$a \cdot \max((1 + kd(1 - \beta))g_k, \beta a^{2k}, d(1 - \beta)a^{2k}) < 1. \tag{13}$$

For $k = 1$, (13) becomes

$$a^3 \cdot \max((1 + d(1 - \beta))/(1 + a), \beta, d(1 - \beta)) < 1,$$

for which the optimal setting of $\beta = (d+1)/(a+d+1)$ leads to the condition that $a^3 < 1 + a/(d+1)$.

Once again, we do not obtain stronger results for higher values of k, as can be seen as follows. Dropping the first term in the maximum expression of (13) and setting β optimally to $\beta = d/(d+1)$ leads to the necessary condition $a^{2k+1} < 1 + 1/d$, which is more stringent than $a^3 < 1 + a/(d+1)$ for $k > 1$ and $d \geq 1$.

4.3 NP-*Complete Problems Other Than Satisfiability*

We now sketch a proof of Corollary 1.

Let A be a language to which satisfiability reduces under a Karp reduction R that runs in time $n^{1+o(1)}$ on a random access machine such that each bit of the reduction can be computed in time $n^{o(1)}$ on a random access machine. Note that the latter computation can be simulated in time $n^{o(1)}$ on a Turing machine with one work tape and random access to the input. Suppose that there is a Turing machine M with a d-dimensional work tape and random access to the input that solves A in time $t(n)$. Then we can construct a Turing machine N of the same type that solves satisfiability as follows: On input x, N simulates M on input $R(x)$ by interleaving the cells of M's work tape by $n^{o(1)}$ auxiliary cells in one dimension of N's work tape. Each time M needs access to a bit of $R(x)$, N uses the $n^{o(1)}$ auxiliary cells around the cell corresponding to N's tape head location to compute the required bit of $R(x)$ from scratch. The resulting simulation N runs in time $t(n^{1+o(1)}) \cdot n^{o(1)}$. The statement of Theorem 2 then rules out that $t(n) \leq n^a$ for constant $a < \sqrt{(d+2)/(d+1)}$.

The proof of Theorem 3 carries over in a similar way.

Acknowledgments

We would like to thank Ravi Kannan, Rahul Santhanam, and the anonymous ICALP referees for helpful discussions, pointers to the literature, and other suggestions.

References

[1] S. Cook. A hierarchy theorem for nondeterministic time complexity. *Journal of Computer and System Sciences*, 7:343–353, 1973.

[2] S. Cook. Short propositional formulas represent nondeterministic computations. *Information Processing Letters*, 26:269–270, 1988.

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

[4] L. Fortnow. Time-space tradeoffs for satisfiability. *Journal of Computer and System Sciences*, 60:337–353, 2000.

[5] L. Fortnow and D. van Melkebeek. Time-space tradeoffs for nondeterministic computation. In *Proceedings of the 15th IEEE Conference on Computational Complexity*, pages 2–13. IEEE, 2000.

[6] F. Hennie. One-tape off-line turing machine computations. *Information and Control*, 8:553–578, 1965.

[7] J. Hopcroft and J. Ullman. Relations between time and tape complexities. *Journal of the ACM*, 15:414–427, 1968.

[8] R. Kannan. Alternation and the power of nondeterminism. In *Proceedings of the 25th ACM Symposium on the Theory of Computing*, pages 344–346. ACM, 1983.

[9] R. Kannan. Towards separating nondeterminism from determinism. *Mathematical Systems Theory*, 17:29–45, 1984.

[10] R. Lipton and A. Viglas. On the complexity of SAT. In *Proceedings of the 40th IEEE Symposium on Foundations of Computer Science*, pages 459–464. IEEE, 1999.

[11] W. Maass, G. Schnitger, E. Szemeredi, and G. Turan. Two tapes versus one for off-line Turing machines. *Computational Complexity*, 3:392–401, 1993.

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

[13] J. Seiferas, M. Fischer, and A. Meyer. Separating nondeterministic time complexity classes. *Journal of the ACM*, 25:146–167, 1978.

[14] D. van Melkebeek. Time-space lower bounds for satisfiability. *Bulletin of the European Association for Theoretical Computer Science*, 73:57–77, 2001.

[15] K. Wagner and G. Wechsung. *Computational Complexity.* Reidel Publishing, 1986.