

# Time-Space Tradeoffs for Counting NP Solutions Modulo Integers

Ryan Williams\*  
*Carnegie Mellon University*

## Abstract

We prove the first time-space tradeoffs for counting the number of solutions to an NP problem modulo small integers, and also improve upon the known time-space tradeoffs for SAT. Let  $m$  be a positive integer, and define MOD $_m$ -SAT to be the problem of determining if a given Boolean formula has exactly  $km$  satisfying assignments, for some integer  $k$ . We prove that for all primes  $p$ , *except for possibly one of them*, MOD $_p$ -SAT is not solvable in  $n^c$  time and  $n^{o(1)}$  space on RAMs, for  $c \geq 1$  satisfying  $c^3 - c^2 - 2c + 1 < 0$  ( $c < 1.801$  suffices). That is, there is at most one prime  $p$  that does not satisfy the lower bound. Note that such a lower bound does not follow from the SAT time-space tradeoffs, as we do not know of an efficient deterministic reduction from SAT to MOD $_p$ -SAT.

The result is non-constructive, in that it does not provide an explicit prime for which the lower bound holds. However, we can prove that the same limitation holds for SAT and MOD $_6$ -SAT, as well as MOD $_m$ -SAT for any composite  $m$  that is not a prime power. Our main tool is a general method for rapidly simulating deterministic RAM computations with restricted space, by counting the number of solutions to NP predicates modulo primes. The simulation converts an ordinary RAM into a “canonical one” that runs in roughly the same amount of time and space, yet its configuration sequences have nice properties suitable for counting.

## 1 Introduction

The class MOD $_k$ P is the collection of languages  $L$  for which there is a nondeterministic polytime algorithm that, on input  $x$ , has  $(0 \bmod k)$  accepting paths if and only if  $x \in L$ . The MOD classes naturally formalize the complexity of counting solutions to problems, modulo small integers. Recent work by Valiant and others [Val04, Val06, CC06, CP07] has brought the problems of counting solutions modulo primes (which we call “MOD $_p$  problems”) back to the forefront of current research. Surprising algorithms and completeness results have been found for particular interesting MOD $_p$  problems.

In this work we consider the diametric problem of proving concrete limitations on how efficiently that MOD $_p$  problems can be solved. Substantial strides have been recently made in establishing deterministic time-space tradeoff lower bounds for nondeterministic time and the polynomial hierarchy—our goal here is to develop methods for extending these arguments to MOD $_p$  problems,

---

\*Supported in part by the NSF ALADDIN Center under grant CCR-0122581, and a grant from Google, Inc.  
Email: ryanw@cs.cmu.edu.

in a general way that should also be useful for proving time-space lower bounds on other types of problems in the future.

At first, it appears that such results might follow without much trouble, given that  $\text{NP}$  can be reduced to  $\text{MOD}_p\text{P}$  (for all primes  $p$ ) by randomized reductions, via the well-known Valiant-Vazirani lemma [VV86]. A quasilinear time version of Valiant and Vazirani's reduction has been found by Naik, Regan, and Sivakumar [NRS95]. Moreover, Toda and Ogihara [TO92] showed that the entire polynomial hierarchy reduces to  $\text{MOD}_p\text{P}$  using two-sided randomized reductions (but for  $\Sigma_k\text{P}$  where  $k \geq 2$ , the best reduction we know of from  $\Sigma_k\text{P}$  to  $\text{MOD}_p\text{P}$  takes  $\Theta(n^{k+1})$  time, cf. Gupta [Gup98]).

However, despite their time-efficiency, the inherent randomness of these reductions is a major difficulty in applying them to obtain time-space lower bounds, since we do not know how to remove the use of randomness even if we assume that  $\text{MOD}_k\text{P}$  problems have efficient algorithms. (We note in passing that, assuming certain unproven circuit lower bounds, deterministic versions of Valiant-Vazirani do exist, cf. [KvM02].)

## 1.1 Our Results

We show how to extend the techniques for proving superlinear time-space tradeoffs for SAT to  $\text{MOD}_p\text{-SAT}$ , for *almost* all primes  $p$ . In particular, for all distinct primes  $p$  and  $q$ , we prove that at least one of  $\text{MOD}_p\text{-SAT}$  and  $\text{MOD}_q\text{-SAT}$  exhibits a time-space tradeoff lower bound, identical to the best known for SAT. Our primary technical contribution is a simulation of time  $T$  and space  $S$  machines that runs in  $(TS)^{1/2+\varepsilon}$  time for all  $\varepsilon > 0$ , by counting modulo two distinct primes.

**Definition 1.1** *Let  $p$  and  $q$  be integers greater than 1. A  $\text{MOD}_p$  machine is a nondeterministic machine with a modified acceptance condition: it accepts iff the number of its accepting computation paths is divisible by  $p$ . A  $\text{MOD}_q\text{MOD}_p$  machine is a  $\text{MOD}_q$  machine equipped with an oracle to some linear time  $\text{MOD}_p$  machine. We define  $\text{MOD}_q\text{MOD}_p\text{TIME}[t(n)]$  to be the class of sets accepted by  $\text{MOD}_q\text{MOD}_p$  machines in  $t(n)$  time.*

**Theorem 1.1 (Speedup by Modular Counting)** *Let  $M$  be a deterministic machine running in time  $T$  and space  $S$ , let  $B(n) \leq T(n)$ , let  $\varepsilon > 0$  be sufficiently small, and let  $p$  and  $q$  be distinct primes. Then there is a  $\text{MOD}_q\text{MOD}_p$  machine  $N$  such that  $L(M) = L(N)$ , whereby  $N$  runs for  $O(B(n)S(n)\log T(n))$  time in its  $\text{MOD}_q$  mode, runs for  $O(\log(B(n)S(n)\log T(n)))$  time in its  $\text{MOD}_p$  mode, then runs in deterministic  $T^{1+\varepsilon}(n)/B(n)$  time and  $S(n)\log T(n)$  space. Moreover, the input to the final deterministic part of the computation has only  $O(n + S(n))$  length.*

In our class notation (cf. Section 2.2), we write the above as:

$$\text{DTISP}[T(n), S(n)] \subseteq (\text{MOD}_q B(n)S(n)\log T(n)) (\text{MOD}_p \log(B(n)S(n)\log T(n)))\text{DTISP}[T^{1+\varepsilon}(n)/B(n), S(n)\log T(n)].$$

By choosing  $B$  such that  $BS\log T = T^{1+\varepsilon}/B$ , it follows that  $\text{DTISP}[T(n), S(n)]$  is contained in  $\text{MOD}_q\text{MOD}_p\text{TIME}[(T(n)S(n))^{1/2+\varepsilon}]$ , for all  $\varepsilon > 0$ .

Using Theorem 1.1 and some elementary number theory, much of the research on time-space tradeoff lower bounds for SAT on RAMs [FLvMV05] can be directly transferred over to  $\text{MOD}_p\text{-SAT}$ . Independently of the  $\text{MOD}_p\text{-SAT}$  transfer arguments, we also add a new argument to this line of work, culminating in a new collection of time-space lower bounds.

**Theorem 1.2** *The following problems require  $\Omega(n^c)$  time on random-access machines using  $n^{o(1)}$  space, for all  $c \in (1, 2)$  such that  $c^3 - c^2 - 2c + 1 < 0$ :*

- SAT, the satisfiability problem for Boolean CNF formulas.
- $\text{MOD}_p\text{-SAT}$ , the problem of counting the number of satisfying assignments to a Boolean formula modulo  $p$ , for all primes  $p$  except for possibly one of them,
- $\text{MOD}_m\text{-SAT}$ , for all integers  $m$  that are not prime powers.

The time lower bound implied by Theorem 1.2 is  $\Omega(n^{1.8019\dots})$ , which is the largest known to date, even for satisfiability. Our lower bounds hold for any NP problem with sufficiently efficient parsimonious reductions from SAT. More details on this point are provided in the preliminaries.

Unfortunately, Theorem 1.2 does not tell us which one of the primes might be the exception in the  $\text{MOD}_p\text{-SAT}$  lower bound. In order for our argument to become constructive, it seems that we would need to prove a hypothesized time hierarchy theorem. Define  $\text{MOD}_k\text{TIME}[t(n)]$  to be the class of languages accepted by a  $\text{MOD}_k$  machine in  $t(n)$  time.

**Hypothesis 1** *For all primes  $p$ , there exists a prime  $q \neq p$  and time constructible  $t(n) > n$  such that  $\text{MOD}_p\text{TIME}[t] \not\subseteq \text{MOD}_q\text{TIME}[o(t)]$ .*

The hypothesis looks very reasonable in light of current knowledge, such as the circuit lower bounds for computing  $\text{MOD}_p$  with OR, AND, NOT, and  $\text{MOD}_q$  gates [Smo87]. It seems extremely counterintuitive that counting solutions modulo one prime would somehow always be faster, if one could count modulo a different prime. If the hypothesis is true, then indeed it follows that  $\text{MOD}_p\text{-SAT}$  requires  $n^{1.8}$  time and  $n^{o(1)}$  space on deterministic RAMs, for all primes  $p$ .

## 2 Preliminaries

We assume familiarity with the usual complexity theoretic notions of time, space, and alternation. All functions used to bound runtime are assumed time constructible, and functions used to bound space usage are space constructible. Define  $\text{DTISP}[t(n), s(n)]$  to be the class of sets accepted by a RAM that runs in  $t(n)$  time and  $s(n)$  space, simultaneously. We shall extensively study the DTISP class in the subpolynomial space ( $s(n) = n^{o(1)}$ ) setting. For convenience, we use the notation  $\text{DTS}[t(n)] := \text{DTISP}[t(n), n^{o(1)}]$ .

Our work relies on a careful analysis of configuration graphs of machines. For completeness, we give a definition. Let  $M$  be a deterministic machine running in time  $T(n)$  and space  $S(n) \geq \log n$ , and let  $x$  be an input string.

**Definition 2.1** *The configuration graph  $G_{M,x}$  of  $M(x)$  has  $2^{cS(n)}$  nodes for a sufficiently large integer  $c > 1$ , where every node of the graph is uniquely labeled by a  $cS(n)$ -bit string. There is an arc from the node labeled  $v$  to the node labeled  $w$  if and only if, when  $v$  and  $w$  are construed as configurations of  $M$ , the configuration  $w$  is obtained by executing  $M$  on  $x$  in configuration  $v$  for one step.*

Since  $M$  is deterministic, observe that the outdegree of any node in  $G_{M,x}$  is at most one. Notice that our definition of configuration graph includes *all* possible strings of length  $cS(n)$  as nodes, so many of the nodes do not correspond to legitimate machine configurations. Those nodes will have indegree and outdegree zero.

## 2.1 Modular Counting Machines

We use an extension of alternating machines introduced by Allender and Gore [AG94], which has not only existential and universal modes but also “MOD $_m$  modes” for various moduli  $m$ . A state in “MOD $_m$  mode” acts as one expects it would, in the following sense. Let us assume that the configuration graph for a given alternating machine is a tree. Let  $\sigma$  be a configuration in “MOD $_m$  mode” where the previous configuration was in a different type of mode (so, an alternation has just occurred). Let  $T$  be the maximal subtree of configurations, rooted at  $\sigma$ , where no alternation takes place. Then  $\sigma$  is accepting iff the number of leaves of  $T$  that are accepting is divisible by  $m$ .

Let us suppose that an alternating machine alternates to specific modes only at specific timesteps, irrespective of the input or current configuration. We say that the *signature* of such a machine is a chronological description of the modes taken during any path of the computation. Typically, the pseudocode for an alternating machine contains commands of the form “*Existentially guess  $x$  of length  $b$* ” and “*Universally guess  $x$  of length  $b$* ”, where  $x$  is some string and  $b$  is a positive integer. (These commands mean that the machine is switched to that particular mode and  $x$  is chosen over all possible strings of length  $b$ .) In a similar manner, our pseudocode for a machine with MOD $_k$  in its signature uses the command “*Modulo  $k$  guess  $x$  of length  $b$* ” analogously.

## 2.2 Class notation

We also use an unorthodox but natural notation for alternating classes that we have found convenient for our arguments. Define  $(\exists f(n))\mathcal{C}$  to be the class of languages recognized by some nondeterministic machine  $N$  that, on input  $x$ , writes a  $O(f(n))$  bit string  $y$  nondeterministically, then feeds the input  $\langle x, y \rangle$  to a machine from class  $\mathcal{C}$ . The classes  $(\forall f(n))\mathcal{C}$  and  $(\text{MOD}_m f(n))\mathcal{C}$  are defined similarly (with co-nondeterministic machines and MOD $_m$  machines, respectively). When a machine recognizing a language in class  $(\exists f(n))\mathcal{C}$  is guessing the  $O(f(n))$  bits, we say that it is *in an existential quantifier*. Similarly, we define being *in a universal quantifier* for  $(\forall f(n))\mathcal{C}$ .

### 2.3 The power of $\text{MOD}_p\text{-SAT}$

Our lower bound for  $\text{MOD}_p\text{-SAT}$  follows the strategy of proving that  $\text{MOD}_p\text{TIME}[n] \not\subseteq \text{DTS}[n^c]$  for a constant  $c > 1$ , arguing that this implies  $\text{MOD}_p\text{-SAT}$  is not in  $\text{DTS}[n^{c-o(1)}]$ . This kind of strategy (exploiting the completeness of a problem) has been invoked in other lower bound arguments as well [FLvMV05, AKRRV01].

**Theorem 2.1** *Let  $k \geq 2$  be an integer. If  $\text{MOD}_k\text{-SAT} \in \text{DTS}[n^c]$ , then  $\text{MOD}_k\text{TIME}[n] \subseteq \text{DTS}[n^{c+o(1)}]$ .*

**Proof.** (Sketch) [FLvMV05, Tou01] give a reduction from an arbitrary  $L \in \text{NTIME}[n]$  to SAT that takes a string  $x$  and converts it to an equivalent formula that is of length at most  $|x|\text{poly}(\log|x|)$ , where each bit of the formula can be computed individually in  $|x|^{o(1)}$  time. From this, the implication

$$\text{SAT is in } n^c \text{ time and } n^{o(1)} \text{ space} \implies \text{NTIME}[n] \subseteq \text{DTS}[n^{c+o(1)}]$$

follows by performing the appropriate reduction and executing the presumed SAT algorithm. We point out that the reduction above is actually *parsimonious*.

From a deterministic linear time predicate  $M$  and input  $x$ , the reduction creates a formula  $\phi_{M,x}(y, z)$ , whereby there is a  $y$  of length  $c|x|$  such that  $M(x, y)$  accepts if and only if the same  $y$  (construed as a variable assignment) along with some  $z$  of length  $c|x|\text{poly}(\log|x|)$  satisfies  $\phi_{M,x}(y, z)$ . Each such valid  $z$  is essentially an encoding of the computation history of the deterministic algorithm  $M$  running on  $(x, y)$ , and is *unique* with respect to a given  $(x, y)$  pair. Thus, there is a 1-1 correspondence between  $y$ 's such that  $M(x, y)$  accepts, and  $(y, z)$  pairs such that  $\phi_{M,x}(y, z)$  is satisfied. Hence the number of valid  $y$  such that  $M(x, y)$  accepts is divisible by  $k$  iff the number of  $(y, z)$  satisfying  $\phi_{M,x}$  is divisible by  $k$ , so the above reduction also serves as a reduction from an arbitrary  $L_k \in \text{MOD}_k\text{TIME}[n]$  to  $\text{MOD}_k\text{-SAT}$ .  $\square$

## 3 Related Work

**Modulo Counting Classes.** The class  $\text{MOD}_k\text{P}$  was introduced by Cai and Hemachandra [CH89]. Beigel and Gill [BG92] showed several closure properties hold for these classes— for example, for all primes  $p$ , the class  $\text{MOD}_p\text{P}$  is closed under union, intersection, and complement. Thus it is strongly believed that  $\text{MOD}_p\text{P} \neq \text{NP}$  for all primes  $p$ .

In contrast to the Valiant-Vazirani lemma and Toda's theorem, which show the power of  $\text{MOD}_k\text{P}$ , Beigel, Buhrman, and Fortnow [BBF98] demonstrated a very interesting oracle collapse/separation for MOD classes. A consequence of their oracle construction is that for all distinct primes  $p$  and  $q$ , there is an oracle  $A$  such that  $\text{P}^A = \text{MOD}_p\text{P}^A$ , yet  $\text{MOD}_q\text{P}^A = \text{NP}^A = \text{EXP}^A$ . That is, there are relativized worlds where for any primes  $p$  and  $q$ , counting NP solutions modulo  $p$  is easy, yet counting NP solutions modulo  $q$  is as hard as exponential time.

**Circuit Lower Bounds with Composite Gates.** There has been substantial work on trying to prove lower bounds for circuits with  $\text{MOD}_6$  gates, and in general, circuits with  $\text{MOD}_{pq}$  gates where  $p$  and  $q$  are distinct primes [BST90, Gro01, CGPT06]. Circuits with such gates appear to be

possibly far more powerful than circuits with merely AND, OR, NOT, and  $\text{MOD}_p$  gates (for some fixed prime  $p$ ). Our work shows the power of  $\text{MOD}_{pq}$ -SAT, as the known proofs of lower bounds for SAT can be extended to it.

**Time-Space Tradeoffs for Nondeterminism.** Using a variant on Nepomnjascii’s theorem [Nep70], Kannan [Kan84] showed that there is a  $k \geq 1$  such that for all  $j \neq k$ ,  $\text{NTIME}[n^j] \not\subseteq \text{DTISP}[n^j, o(n^{j/k})]$ , thus establishing a time-space tradeoff for nondeterminism. This line of study was revived in the late 90’s by Fortnow [For97], who proved that  $\text{SAT} \notin \text{DTIME}[n^{1+o(1)}] \cap \text{NL}$ . Fortnow-Lipton-Viglas-Van Melkebeek [FLvMV05] sharpened the tools and arguments, showing that SAT is not in  $n^{\phi-\varepsilon}$  time and  $n^{o(1)}$  space for all  $\varepsilon > 0$ , where  $\phi$  is the golden ratio. Earlier work of ours [Wil05] improved the time lower bound to greater than  $n^{\sqrt{3}}$ . Building on our argument, Diehl and Van Melkebeek [DvM06] gained a slight improvement to  $n^{1.759}$ .

**Lower Bounds for the Counting Hierarchy.** Several interesting lower bounds on counting problems have been discovered in the past. Allender and Gore [AG94] proved that uniform  $\text{ACC}^0$  is properly contained in PP. Caussinus *et al.* [CMTV98] showed that uniform  $\text{ACC}^0$  is properly contained in MODPH (a counting version of the polynomial hierarchy). Allender [All98] showed that the Permanent is not in  $\text{TC}^0$ , thus  $\text{TC}^0 \neq \text{PP}$ . Allender *et al.* [AKRRV01] proved time-space tradeoffs for MAJ-MAJ-SAT (MAJ-MAJ-SAT is a complete problem in the second level of the counting hierarchy, a generalization of MAJ-SAT), showing that it is not contained in unbounded probabilistic time  $n^{1+o(1)}$  and  $n^\delta$  space for all  $\delta < 1$ .

## 4 Intuition Behind This Work

The manner in which our results are proved is perhaps more interesting than the results themselves. At the heart of our work is the Speedup By Modular Counting Theorem (Theorem 1.1), which gives a new method for simulating deterministic computations with restricted time and space, via counting. We first show that every machine  $M$  has an equivalent “canonical” version  $\hat{M}$ . The canonical version uses roughly the same time and space as  $M$ ; however, the task of counting particular properties of  $\hat{M}$ ’s configurations is much easier. The canonical machine follows from an efficient reversible simulation of irreversible computation, first given by Bennett [Ben89]. The simulation of  $\hat{M}$  on an input  $x$  counts the number of certain objects (which we call “complete configuration sequences with a number of mistakes divisible by  $p$ ” for a prime  $p$ ) efficiently with a  $\text{MOD}_p\text{TIME}[n]$  oracle, such that the difference in the number of such objects counted in the accepting and rejecting cases is congruent to 1 *modulo*  $q$ , for prime  $q \neq p$ . Thus in principle, one can tell the difference between the accepting and rejecting cases by counting modulo  $q$ . Finally, we show that one can determine very efficiently (prior to simulation) which modulus it should “expect” in the accepting case versus the rejecting case, and therefore discern between the two cases.

Theorem 1.1 (and extensions of it) allows us to transfer arguments establishing time-space tradeoffs for nondeterminism directly to counting solutions modulo primes, by substituting nondeterminism with “counting mod  $p$ ”, and conondeterminism with “counting mod  $q$ ”. For example,

Lipton and Viglas [LV99] proved an  $\Omega(n^{\sqrt{2}-\varepsilon})$  time lower bound for solving SAT on  $n^{o(1)}$ -space RAMs. One way to phrase their argument uses the fact that  $\text{DTISP}[t, s] \subseteq \Sigma_2\text{TIME}[(ts)^{1/2}]$  in the following way: assuming  $\text{NTIME}[n] \subseteq \text{DTS}[n^c]$ , one can prove

$$\Sigma_2\text{TIME}[t] \subseteq \text{NTIME}[t^c] \subseteq \text{DTS}[t^{c^2}] \subseteq \Sigma_2\text{TIME}[t^{c^2/2+o(1)}]$$

for a given time bound  $t$ . The first two inclusions follow by assumption, and the last inclusion follows from the fact. For  $c^2 < 2$ , the above inclusion is a contradiction. An analogous lower bound for  $\text{MOD}_6\text{TIME}[n]$  can be obtained by assuming  $\text{MOD}_6\text{TIME}[n] \subseteq \text{DTS}[n^c]$ , and proving for all  $\varepsilon > 0$  that

$$\begin{aligned} (\text{MOD}_3 t)(\text{MOD}_2 t)\text{DTS}[t] &\subseteq (\text{MOD}_3 t)\text{DTS}[t^c] \\ &\subseteq \text{DTS}[t^{c^2}] \\ &\subseteq (\text{MOD}_3 t^{c^2/2+\varepsilon})(\text{MOD}_2 t^{c^2/2+\varepsilon})\text{DTS}[t^{c^2/2+\varepsilon}]. \end{aligned}$$

The first inclusion follows from the fact that  $\text{MOD}_2\text{TIME}[n] \subseteq \text{MOD}_6\text{TIME}[n] \subseteq \text{DTS}[n^c]$ , the second from  $\text{MOD}_3\text{TIME}[n] \subseteq \text{MOD}_6\text{TIME}[n] \subseteq \text{DTS}[n^c]$ , and the third from the Speedup by Modular Counting Theorem (Theorem 1.1). Again, for  $c^2 < 2(1-\varepsilon)$ , we have a contradiction. The above kind of transfer can be carried out on other time-space tradeoff arguments similarly.

## 5 Canonical Machines

We begin by discussing the notion of a canonical machine. For our simulations of DTS, we require the machine being simulated to have a configuration graph of a very regular type, which we call *canonical*. It turns out that a deterministic small-space machine can be converted into a canonical one, without a significant increase in time and space usage.

**Definition 5.1** *A machine  $M$  is **canonical** iff for all inputs  $x$ , every node in the configuration graph  $G_{M,x}$  has outdegree and indegree exactly one. That is, the graph is a union of disjoint cycles along with isolated nodes having self-loops.*

By definition, a canonical machine does not halt, since no configuration has outdegree zero. Thus we need to modify the acceptance condition for a canonical machine.

**Definition 5.2** *Let  $M$  be canonical.  $M$  **accepts input**  $x$  if and only if  $M(x)$  (started in its initial configuration) reaches a configuration with an “accept” state before it reaches its initial configuration again.*

A canonical machine is obviously deterministic, but it also enjoys the following useful property.

**Proposition 1** *Let  $M$  be canonical, let  $c$  be a node of  $G_{M,x}$ , and let  $t$  be a non-negative integer. Then there are unique configurations  $c'_t$  and  $c''_t$  such that, when  $M(x)$  is executed from  $c'_t$  for  $t$  steps,  $M(x)$  ends in configuration  $c$ , and when  $M(x)$  is executed from  $c$  for  $t$  steps,  $M(x)$  ends in configuration  $c''_t$ .*

Proposition 1 *does not hold* for deterministic machines in general, as there can be numerous configurations that lead to a common configuration by executing each one for  $t$  steps, and the number of such configurations can, in principle, depend upon the given input. The “unique predecessor” and “unique successor” properties of canonical machines shall prove to be indispensable in our later simulations of space-bounded computation.

## 5.1 Making machines canonical

In order to show that every deterministic machine can be turned into an equivalent canonical one, we apply a theorem of Bennett that shows how to convert any deterministic machine into one that is *reversible*, with negligible penalty to its time and space complexity. Recall the definition of *reversible machine*: a deterministic machine is reversible if and only if its configuration graph on all inputs is such that all nodes have indegree at most one and outdegree at most one.

**Theorem 5.1 (Bennett [Ben89])** *For every deterministic machine  $M$  running in time  $T(n)$  and space  $S(n)$ , and every integer  $k \geq 1$ , there is a reversible machine  $M'$  that runs in  $T^{1+1/k}(n)$  time and  $2^k S(n) \log_2 T(n)$  space, such that  $L(M) = L(M')$ .*

Along with Bennett [Ben89], see also Buhrman, Tromp, and Vitanyi [BTV01] for an exposition of the proof. We require two specific properties of the reversible  $M'$  obtained in Theorem 5.1.

**Remark 1** *The reversible  $M'$  of Theorem 5.1 has the following additional properties:*

1. *For all positive integers  $n$ , there is a unique configuration  $A_n$  that can be computed in  $O(S(n))$  time such that, on all  $x$  of length  $n$ ,  $M'(x)$  accepts  $\iff M'(x)$  is in configuration  $A_n$  in precisely  $T(n)$  steps from its initial configuration.*
2. *There is a machine  $M'_R$  such that on all  $x$ ,  $G_{M'_R, x}$  is equivalent to  $G_{M', x}$  except that all arcs point in the opposite directions. That is,  $M'$  on  $x$  goes from configuration  $C$  to  $C'$  in one step if and only if  $M'_R$  on  $x$  goes from configuration  $C'$  to  $C$  in one step.*

Applying the above two remarks, it is not difficult to construct an equivalent canonical machine for any deterministic machine that has the same asymptotic time and space complexity as the reversible simulation of Theorem 5.1.

**Theorem 5.2** *Let  $M$  be a deterministic machine running in time  $T(n)$  and space  $S(n)$ . For all  $\varepsilon > 0$ , there exists a canonical machine  $\hat{M}$  that runs in  $T^{1+\varepsilon}(n)$  time and  $2^{O(1/\varepsilon)} S(n) \log_2 T(n)$  space, and  $L(M) = L(\hat{M})$ .*

**Proof.** The first step is to apply Theorem 5.1 with  $k > 1/\varepsilon$ , turning  $M$  into an equivalent reversible machine  $M'$ . Let  $M'_R$  be the machine guaranteed by Remark 1. Since  $M'$  is reversible,  $M'_R$  is reversible as well. Thus the graphs  $G_{M', x}$  and  $G_{M'_R, x}$  are already such that the indegrees and outdegrees are at most one. To obtain an equivalent  $\hat{M}$  so that every node of some  $G_{\hat{M}, x}$  has indegree and outdegree *exactly* one, we make the following change.

$\hat{M}$  has two *modes*: forward and backward.  $\hat{M}$  starts in forward mode from the initial configuration of  $M'$ , and simulates  $M'$  normally. Two rules are applied:

- If  $\hat{M}$  is in forward mode and reaches a configuration where no transition applies, it switches to backward and starts executing  $M'_R$ .
- If  $\hat{M}$  is in backward mode and reaches a configuration where no transition applies, it switches to forward and starts executing  $M'$ .

We claim that  $\hat{M}$  is canonical. It suffices for us to prove that every node in a configuration graph  $G_{\hat{M},x}$  has indegree and outdegree exactly one. Notice that  $G_{\hat{M},x}$  has precisely double the nodes of  $G_{M',x}$ , one for each of the two modes of  $\hat{M}$ . Thus we can label each node of  $G_{\hat{M},x}$  as a pair  $\langle C, m \rangle$ , where  $C$  is a configuration of  $G_{M',x}$  and  $m$  is a mode. Therefore  $\hat{M}(x)$  has precisely twice the number of configurations as  $M'(x)$ .

Let  $\langle C, m \rangle$  be a configuration of  $G_{\hat{M},x}$ . There are four possible cases.

1.  **$C$  has outdegree 0 and indegree 0.** (That is,  $C$  is not a legal machine configuration.) Then the edges adjacent to  $\langle C, \text{forward} \rangle$  and  $\langle C, \text{backward} \rangle$  in  $G_{\hat{M},x}$  are  $(\langle C, \text{forward} \rangle, \langle C, \text{backward} \rangle)$  and  $(\langle C, \text{backward} \rangle, \langle C, \text{forward} \rangle)$ .
2.  **$C$  has outdegree 1 and indegree 1 in  $G_{M',x}$ .** Let  $(C, C')$  and  $(C'', C)$  be the respective edges. Then  $(\langle C, \text{forward} \rangle, \langle C', \text{forward} \rangle)$  and  $(\langle C'', \text{forward} \rangle, \langle C, \text{forward} \rangle)$  are the edges adjacent to  $\langle C, \text{forward} \rangle$  in  $G_{\hat{M},x}$ . Similarly,  $\langle C, \text{backward} \rangle$  has edges  $(\langle C, \text{backward} \rangle, \langle C', \text{backward} \rangle)$ , and  $(\langle C'', \text{backward} \rangle, \langle C, \text{backward} \rangle)$ .
3.  **$C$  has outdegree 0 and indegree 1.** Let  $(C, C')$  be the respective edge. Then the edges adjacent to  $\langle C, \text{forward} \rangle$  in  $G_{\hat{M},x}$  are  $(\langle C, \text{forward} \rangle, \langle C', \text{forward} \rangle)$  and  $(\langle C, \text{backward} \rangle, \langle C, \text{forward} \rangle)$ . Similarly,  $(\langle C, \text{backward} \rangle, \langle C, \text{forward} \rangle)$  and  $(\langle C', \text{backward} \rangle, \langle C, \text{backward} \rangle)$  are the edges adjacent to  $\langle C, \text{backward} \rangle$  in  $G_{\hat{M},x}$ .
4.  **$C$  has outdegree 1 and indegree 0.** Analogous to the previous case.

Finally, it is easy to see that  $M(x)$  accepts in  $T(|x|)$  time if and only if  $\hat{M}(x)$  accepts in  $T(|x|)$  time.  $\square$

## 6 Simulating Space-Bounded Machines By Counting

In this section, we establish Theorem 1.1, which shows that any language in  $\text{DTISP}[T(n), S(n)]$  can be simulated extremely efficiently by a machine of signature  $\text{MOD}_p \text{MOD}_q$ , for any distinct primes  $p$  and  $q$ . To do this, we first review a method for simulating any time  $T(n)$ , space  $S(n)$  machine by a  $\Sigma_2$  machine that runs in  $(T(n) \cdot S(n))^{1/2}$  time [Nep70, Kan84]. That is, one can “speed up” a deterministic machine by introducing two alternations in the computation, when the machine’s time-space product is small. We sketch the construction here for completeness.

**Theorem 6.1 (Follows from Nepomnjascii [Nep70], Kannan [Kan84])**

$$\text{DTISP}[T(n), S(n)] \subseteq \Sigma_2 \text{TIME}[(T(n)S(n))^{1/2}].$$

In particular, for every constructible  $B(n)$ ,

$$\text{DTISP}[T(n), S(n)] \subseteq (\exists B(n)S(n))(\forall \log(B(n)S(n)))\text{DTISP}[T(n)/B(n), S(n)].$$

**Proof.** (Sketch) Let  $M$  be a random access machine running in  $T(n)$  time and  $S(n)$  space. Its simulation  $N(x)$  begins by existentially guessing a sequence of  $B(|x|)$  configurations of  $M(x)$ . It then appends the initial configuration to the beginning of the sequence and the accepting configuration to the end of the sequence. Finally,  $N(x)$  universally guesses an integer  $i \in \{0, \dots, B(|x|)\}$  and simulates  $M(x)$  starting from the  $i$ th configuration in the sequence, accepting if and only if the  $(i + 1)$ th configuration in the sequence is reached in  $T(|x|)/B(|x|)$  steps. It is easy to see that for all  $x$ ,  $M(x)$  accepts if and only if  $N(x)$  accepts. Observe that for  $B(n) = \sqrt{T(n)/S(n)}$ , we have  $\text{DTISP}[T(n), S(n)] \subseteq \Sigma_2 \text{TIME}[(T(n)S(n))^{1/2}]$ .  $\square$

We shall prove that, with some modifications, a simulation akin to the above can successfully simulate canonical machines when the *existential modes* of  $N$  are replaced by  $\text{MOD}_p$  modes, and the *universal modes* are replaced by  $\text{MOD}_q$  modes, where  $p$  and  $q$  are distinct primes. Theorem 1.1 follows from such a simulation. We begin with some definitions. Let machine  $M$  run in time  $T(n)$  in the following.

**Definition 6.1** A  $B$ -configuration sequence for  $M$  on  $x$  is a tuple  $\langle C_0, C_1, \dots, C_B, C_{B+1} \rangle$ , where  $C_i$  are configurations of  $M$  on  $x$ .

Such a sequence is called **complete** if  $C_0$  is the unique initial configuration of  $M$  and  $C_{B+1}$  is the unique accepting configuration.

Such a sequence is **correct** if for all  $i = 0, \dots, B-1$ , when  $M(x)$  is run for exactly  $\lfloor T(|x|)/(B+1) \rfloor + 1$  steps from  $C_i$ , the resulting configuration is  $C_{i+1}$ , and when  $M(x)$  is run for  $T(|x|) - B \cdot (\lfloor T(|x|)/(B+1) \rfloor + 1)$  steps from  $C_B$ , the resulting configuration is  $C_{B+1}$ .

If a  $B$ -configuration sequence is not correct then it is said to have a **mistake at the pair**  $(C_i, C_{i+1})$  when  $C_i$  does not result in  $C_{i+1}$  in  $T(|x|)/(B+1)$  steps.

Note that

$$T(|x|) - (\lfloor T(|x|)/(B+1) \rfloor + 1) \leq T(|x|) - B \cdot T(|x|)/(B+1) = T(|x|)/(B+1) \leq \lfloor T(|x|)/(B+1) \rfloor + 1,$$

so the last pair  $(C_B, C_{B+1})$  requires no more steps than the other pairs. Also, observe the number of  $B$ -configuration sequences depends solely on the space complexity of the machine.

**Claim 1** Let  $n$  and  $B$  be positive integers, and let  $M$  be a machine running in space  $S(n)$ . There is a number  $N(n)$  such that for all  $x$  satisfying  $|x| = n$ , the number of  $B$ -configuration sequences for  $M(x)$  is precisely  $N(n)$ .

**Proof.** Without loss of generality, the number of configurations for  $M$  on inputs of length  $n$  is  $2^{cS(n)+d}$  for some integers  $c > 1$  and  $d > 1$  that are independent of  $n$ . Thus the total number of  $B$ -configuration sequences is  $N(n) = 2^{B(cS(n)+d)}$ .  $\square$

Clearly, for any input  $x$ , the number of  $B$ -configuration sequences that are correct and complete is either zero ( $M(x)$  rejects) or one ( $M(x)$  accepts). The next lemma shows that, for canonical machines, the number of configuration sequences that are merely correct (but not necessarily complete) depends solely on the input length:

**Lemma 6.1** *Let  $\hat{M}$  be canonical and let  $C(n)$  be the number of configurations of  $\hat{M}$  on inputs of length  $n$ . Let  $x$  be an input and  $n = |x|$ . Then for every integer  $B$ , the number of correct  $B$ -configuration sequences for  $\hat{M}(x)$  is exactly  $C(n)$ . In fact, a correct  $B$ -configuration sequence is completely determined by specifying an integer  $i \in \{0, 1, \dots, B + 1\}$  and a configuration  $c$ .*

**Proof.** For each configuration, there is exactly one correct  $B$ -configuration sequence that starts with that configuration, by Proposition 1. Moreover, Proposition 1 implies that, by specifying even a single configuration  $c$  and its position in a correct  $B$ -configuration sequence, the rest of the correct  $B$ -configuration sequence is completely determined, as all configurations prior to  $c$  and all configurations after  $c$  are unique.  $\square$

We now give a way to differentiate between the accepting and rejecting cases for a canonical machine, by counting the mistakes in configuration sequences. The Counting Lemma proves that on any input  $x$ , the number of complete  $B$ -configuration sequences with  $k$  mistakes at adjacent pairs is *independent* of the given input: that is, the number of such sequences depends only on the size of the input  $n$ , the number of configurations  $B$  in the sequence, and whether or not the input leads to acceptance.

**Lemma 6.2 (Counting Lemma)** *Let  $n$  and  $B \geq 1$  be positive integers, let  $k = 0, 1, \dots, B + 1$ , and let  $\hat{M}$  be canonical. Then there are positive integers  $N_A(n, k, B)$  and  $N_R(n, k, B)$  such that, for all inputs  $x$  of length  $n$ :*

1. *If  $\hat{M}(x)$  accepts, then the number of complete  $B$ -configuration sequences for  $\hat{M}(x)$  with exactly  $k$  mistakes (at adjacent pairs) is  $N_A(n, k, B)$ , and*
2. *If  $\hat{M}(x)$  rejects, then the number of complete  $B$ -configuration sequences for  $\hat{M}(x)$  with exactly  $k$  mistakes (at adjacent pairs) is  $N_R(n, k, B)$ .*
3.  $N_A(n, k, B) - N_R(n, k, B) = (-1)^k \binom{B+1}{k}$ .

**Proof.** Fix a canonical  $\hat{M}$  and input  $x$  of length  $n$ . We count the number of ways that one can choose a complete  $B$ -configuration sequence with precisely  $k$  mistakes at adjacent pairs.

We begin with the case  $k = 0$ . Independently of  $B$ ,  $N_A(n, 0, B) = 1$  and  $N_R(n, 0, B) = 0$ , since for any  $B$  there is exactly one correct and complete  $B$ -configuration sequence on an accepted input, and there is no such sequence on a rejected input.

For the remainder of the proof, we assume  $k \geq 1$ . First, before choosing the configurations themselves, we choose the points for which the  $k$  mistakes occur. As there are  $B + 1$  possible pairs for which a mistake can occur, the number of possible choices is  $\binom{B+1}{k}$ .

Suppose the indices  $i_1, \dots, i_k \in \{0, \dots, B\}$  are chosen, with  $i_1 < i_2 < \dots < i_k$ . Next, we need to count the number of complete  $B$ -configuration sequences with mistakes precisely at the

adjacent pairs  $(C_{i_j}, C_{i_{j+1}})$ , for  $j = 1, \dots, k$ . By definition, from configuration  $C_0$  to  $C_{i_1}$ , there is no mistake in the configuration sequence; similarly there is no mistake from  $C_{i_1+1}$  to  $C_{i_2}$ , from  $C_{i_2+1}$  to  $C_{i_3}$ ,  $\dots$ , from  $C_{i_{k-1}+1}$  to  $C_{i_k}$ , and from  $C_{i_k+1}$  to  $C_{B+1}$ . That is, the configuration sequences from  $C_0$  to  $C_{i_1}$ ,  $C_{i_j+1}$  to  $C_{i_{j+1}}$ , and  $C_{i_k+1}$  to  $C_{B+1}$ , are all subsequences of a correct  $B$ -configuration sequence, for  $j = 1, \dots, k-1$ . By Lemma 1, if we specify one configuration at a certain position in a correct configuration sequence, then the entire sequence is determined. Hence we only need to specify the configurations  $C_{i_j}$  for  $j = 2, \dots, k$ , in order to uniquely specify a complete configuration sequence with  $k$  mistakes at exactly the  $(C_{i_j}, C_{i_{j+1}})$  pairs (note  $C_{i_1}$  and  $C_{i_k+1}$  are already uniquely determined by the initial configuration  $C_0$  and accepting configuration  $C_{B+1}$ , respectively).

The counting problem has now boiled down to the question: *fixing the indices  $(i_1, \dots, i_k)$ , how many  $(k-1)$ -tuples of configurations  $(C_{i_2}, C_{i_3}, \dots, C_{i_k})$  are there, such that the complete  $B$ -configuration sequence with  $k$  mistakes uniquely determined by this tuple has a mistake at exactly the pairs  $(C_{i_j}, C_{i_{j+1}})$ ?*

As might be expected, the answer depends on whether or not  $\hat{M}(x)$  accepts or rejects. Let  $N_A(n, k)$  be the number of above tuples for  $\hat{M}(x)$ , on the condition that  $\hat{M}(x)$  accepts. Define  $N_R(n, k)$  similarly, on the condition that  $\hat{M}(x)$  rejects. Recalling that there are  $\binom{B+1}{k}$  ways to pick the indices for the mistakes,

$$\begin{aligned} N_A(n, k, B) &= \binom{B+1}{k} N_A(n, k) \\ N_R(n, k, B) &= \binom{B+1}{k} N_R(n, k), \end{aligned} \quad (*)$$

provided that  $N_R(n, k)$  and  $N_A(n, k)$  actually exist (*i.e.* that these numbers are indeed dependent only on  $n$  and  $k$ ).

The existence of  $N_A(n, k)$  and  $N_R(n, k)$  is proved by induction on  $k$ . Since we assume  $k \geq 1$ , the base case is  $k = 1$ :

- $N_A(n, 1) = 0$ . That is, for fixed  $i_1$ , the number of complete  $B$ -configuration sequences with only one mistake is *zero*, if  $\hat{M}(x)$  accepts. This is because if  $\hat{M}(x)$  accepts, and there is no mistake from the initial configuration  $C_0$  to  $C_{i_1}$ , then there is no mistake from  $C_{i_1}$  to the accept configuration  $C_{B+1}$ ; moreover, if the initial configuration does not lead to a configuration  $C_{i_1}$  without a mistake, then  $C_{i_1}$  does not lead to the accept configuration without a mistake. (Thus the number of mistakes there can be is either zero, or greater than one.)
- $N_R(n, 1) = 1$ . This is because if  $\hat{M}(x)$  rejects, and the initial configuration leads to a  $C_{i_1}$  (uniquely determined by  $i_1$ ) without a mistake, then the accept configuration must lead back to some  $C_{i_1+1}$  such that there is a mistake at  $(C_{i_1}, C_{i_1+1})$ . The point at which this mistake happens is determined by  $i_1$ .

Let  $C(n)$  be the number of configurations of  $\hat{M}$  on inputs of length  $n$ . To complete the characterization of  $N_A(n, k)$  and  $N_R(n, k)$ , we prove for all  $k \geq 2$  that

$$\begin{aligned} N_A(n, k) &= (C(n) - 1)^{k-1} - N_A(n, k-1) \quad (**) \\ N_R(n, k) &= (C(n) - 1)^{k-1} - N_R(n, k-1). \end{aligned}$$

The proof of (\*\*) is as follows. There are  $C(n)^{k-1}$  possible  $(k-1)$ -tuples of configurations. Out of these, there are precisely  $(C(n)-1)^{k-1}$  different  $(k-1)$ -tuples with the property that its corresponding  $B$ -configuration sequence has a mistake at the pairs  $(C_{i_j}, C_{i_{j+1}})$ , for all  $i = 1, \dots, k-1$ . That is,  $(C(n)-1)^{k-1}$  of the possible complete  $B$ -configuration sequences have a mistake at the first  $k-1$  pairs of indices. (This is due to Proposition 1: for every configuration  $C_{i_j}$ , there are  $C(n)-1$  configurations that it does not lead to, for any prescribed number of steps.) However, this quantity is overcounting—we need to subtract those tuples that lead to configuration sequences with a mistake at the first  $k-1$  pairs, but no mistake at the pair  $(C_{i_k}, C_{i_{k+1}})$ .

If the pair  $(C_{i_{k-1}}, C_{i_k})$  does not have a mistake, then  $C_{i_k}$  is uniquely determined by  $C_{i_{k-1}}$ , by Proposition 1. Therefore, the number of  $(k-1)$ -tuples corresponding to complete  $B$ -configuration sequences with mistakes at  $(C_{i_j}, C_{i_{j+1}})$  for all  $j = 1, \dots, k-1$ , but no mistake at  $(C_{i_k}, C_{i_{k+1}})$ , is equal to the number of  $(k-2)$ -tuples  $(C_{i_2}, \dots, C_{i_{k-1}})$  corresponding to complete  $B$ -configuration sequences with mistakes at the indices  $(i_1, \dots, i_{k-1})$ . This number is  $N_A(n, k-1)$  in the accepting case and  $N_R(n, k-1)$  in the rejecting case, by induction.

Finally, we conclude from (\*) and (\*\*) that when  $k \geq 2$  we have

$$\begin{aligned}
N_A(n, k, B) - N_R(n, k, B) &= \binom{B+1}{k} (N_A(n, k) - N_R(n, k)) \text{ by } (*) \\
&= \binom{B+1}{k} (-1) (N_A(n, k-1) - N_R(n, k-1)) \text{ by } (**) \\
&= \binom{B+1}{k} (-1)^2 (N_A(n, k-2) - N_R(n, k-2)) \text{ by } (**) \\
&= \binom{B+1}{k} (-1)^{k-1} (N_A(n, 1) - N_R(n, 1)) \text{ by } (**) \\
&= \binom{B+1}{k} (-1)^{k-1} (-1) = \binom{B+1}{k} (-1)^k.
\end{aligned}$$

□

The Counting Lemma gives exact values for the number of mistakes in configuration sequences, and a clean characterization of how these numbers *differ* in the accepting and rejecting cases. One might ask if we can simulate a canonical machine by simply counting, over all complete  $B$ -configuration sequences, all the mistakes made by adjacent pairs. By choosing  $B = (T(n)/S(n))^{1/2}$ , such a simulation could be implemented in roughly  $(T(n)S(n))^{1/2}$  time as a #P function using an argument like that of Theorem 6.1, yielding a quadratic speedup of a time  $T(n)$ , space  $S(n)$  machine by a #P function. However, one can prove that the Counting Lemma implies

$$\sum_{i=0}^{B+1} i \cdot N_A(n, i, B) = \sum_{i=0}^{B+1} i \cdot N_R(n, i, B).$$

That is, the total number of mistakes committed by all adjacent pairs over all complete  $B$ -configuration sequences is the *same* in the accepting and rejecting cases, so the above proposal does not work. Instead, we count the number of complete  $B$ -configuration sequences whose number of mistakes is *divisible by a prime  $p$* , and compute this number of sequences modulo another prime  $q$ .

The next lemma shows that, in order to distinguish between the accepting and rejecting case, it is enough to perform these counts of sequences and their mistakes modulo two distinct primes.

**Lemma 6.3** *Fix a canonical machine  $\hat{M}$ . Let  $p$  and  $q$  be distinct primes, and let  $B = q^\ell - 1$  for some integer  $\ell > 0$ . Then the number (modulo  $q$ ) of complete  $B$ -configuration sequences whose number of mistakes is divisible by  $p$ , is different in the accepting and rejecting cases. In particular,*

$$\sum_{k=0}^{\lfloor (B+1)/p \rfloor} (N_A(n, kp, B) - N_R(n, kp, B)) \equiv 1 \pmod{q}.$$

**Proof.** Let  $B = q^\ell - 1$  for some  $\ell > 0$ . By the Counting Lemma, the difference in the number of complete  $B$ -configuration sequences with a number of mistakes congruent to 0 mod  $p$  is

$$\begin{aligned} \sum_{k=0}^{\lfloor (B+1)/p \rfloor} (N_A(n, kp, B) - N_R(n, kp, B)) &= \sum_{k=0}^{\lfloor (B+1)/p \rfloor} (-1)^{kp} \binom{B+1}{kp} \\ &= \sum_{k=0}^{\lfloor q^\ell/p \rfloor} (-1)^{kp} \binom{q^\ell}{kp}. \end{aligned}$$

Now we analyze the terms of the form  $\binom{q^\ell}{kp}$ . When  $k = 0$ , then of course  $(-1)^{kp} \binom{q^\ell}{kp} = (-1)^0 \binom{q^\ell}{0} = 1$ . We claim that for all integers  $k$  satisfying  $0 < k \leq \lfloor q^\ell/p \rfloor$  we have  $\binom{q^\ell}{kp} \equiv 0 \pmod{q}$ . The lemma follows immediately, as the claim implies

$$\begin{aligned} \sum_{k=0}^{\lfloor (B+1)/p \rfloor} (N_A(n, kp, B) - N_R(n, kp, B)) &\equiv \sum_{k=0}^{\lfloor q^\ell/p \rfloor} (-1)^{kp} \binom{q^\ell}{kp} \pmod{q} \\ &\equiv 1 \pmod{q}. \end{aligned}$$

We now prove the claim. By elementary combinatorics,

$$kp \cdot \binom{q^\ell}{kp} = q^\ell \cdot \binom{q^\ell - 1}{kp - 1}.$$

(Both sides count the number of ways that one can choose  $kp$  elements from a collection of  $q^\ell$ , and place a mark on one of the  $kp$  elements.) Consider the term  $q^\ell$  on the right-hand side. Since  $0 < k \leq \lfloor q^\ell/p \rfloor$ , we know  $0 < kp \leq \lfloor q^\ell/p \rfloor p < q^\ell$ , since  $p \neq q$  are primes. Therefore, the two terms on the left-hand side and the two terms on the right-hand side are all positive integers, and the equality exhibits two factorizations of the same positive integer. Since  $kp < q^\ell$ , there is a non-trivial factor of  $q^\ell$  that divides  $\binom{q^\ell}{kp}$ , by unique factorization. This proves the claim.  $\square$

Lemma 6.3 and the Counting Lemma allow for the possibility for us to use any distinct primes  $p$  and  $q$  to perform a fast simulation of  $\text{DTISP}[T(n), S(n)]$  using  $\text{MOD}_p$  and  $\text{MOD}_q$  modes. Still, in order to simulate correctly, we need to efficiently compute a modulus that we “expect” to see in the accepting case. The next lemma accomplishes this.

**Lemma 6.4** Fix a canonical machine  $\hat{M}$  that uses  $S(n)$  space. Let  $S(n)$  be constructible in  $O(S(n))$  space and  $T'(n)$  time. For any positive integers  $B$ ,  $p$ , and  $q$ , the quantity

$$\text{MISTAKES}_A(n, B, p, q) := \left( \sum_{k=0}^{\lfloor (B+1)/p \rfloor} N_A(n, kp, B) \right) \pmod q$$

can be computed in  $O(B + T'(n))$  time and  $O(B)$  space. That is, we can compute what the number (modulo  $q$ ) of complete  $B$ -configuration sequences whose number of mistakes is divisible by  $p$  would be on an  $n$ -bit input, if  $\hat{M}(x)$  accepted.

**Proof.** (Sketch) First, compute the total number of possible configurations  $C(n) \pmod q$ . As  $C(n) = 2^{cS(n)}$  for some constant  $c > 1$ , this can be computed directly from the quantity  $cS(n)$ . Assuming  $S(n)$  is constructible in  $T'(n)$  time,  $C(n) \pmod q$  can be computed in  $O(T'(n))$  time.

By following the proof of the Counting Lemma, given  $(C(n) \pmod q)$  one can easily construct each  $(N_A(n, kp, B) \pmod q)$  inductively, in  $O(B)$  time and space. For example, in order to compute

$$E(k) = \binom{B+1}{k} \pmod q = \frac{(B+1) \cdot B \cdots (B+1 - (k+1))}{k \cdot (k-1) \cdots 2} \pmod q,$$

we first compute  $E(1)$  in  $O(1)$  time, and each  $E(i)$  can be computed from  $E(i-1)$  in  $O(1)$  time. Given the values for  $(N_A(n, kp, B) \pmod q)$ ,  $\text{MISTAKES}_A(n, B, p, q)$  is easily constructed in  $O(B)$  time.  $\square$

We finally arrive at the proof of Theorem 1.1.

**Proof of Theorem 1.1.** Given  $M$  that runs in time  $T(n)$  and space  $S(n)$ , let  $\hat{M}$  be the efficient canonical version of  $M$  guaranteed by Theorem 5.2 that runs in time  $T'(n) = T^{1+\varepsilon}(n)$  and space  $S'(n) = S(n) \log T(n)$ . Let  $I_n$  be the unique initial configuration for  $\hat{M}$  on inputs of length  $n$ . We may also assume that  $\hat{M}$  has a unique accepting configuration: by Remark 1, the reversible version  $M'$  of  $M$  has a unique accepting configuration  $A_n$  on inputs of length  $n$ . Then define  $\hat{A}_n = \langle A_n, \text{forward} \rangle$  to be the unique accepting configuration for  $\hat{M}$ .

Informally, the idea is to run the  $\Sigma_2$  simulation of  $\hat{M}$  from Theorem 6.1, but we replace the existential mode with a “ $\text{MOD}_q$  mode” and the universal mode with a “ $\text{MOD}_p$  mode”. We invoke the Counting Lemma to prove correctness.

Fix an input  $x$  and canonical  $\hat{M}$  that runs in  $T'(|x|)$  time on  $x$ . Let  $\ell$  be the smallest integer such that  $B(|x|) \leq q^\ell$ .

We now describe the simulation of  $\hat{M}$  by a  $\text{MOD}_q \text{MOD}_p$  machine:

- (0) Set  $B' = q^\ell - 1$ . Compute  $K = \text{MISTAKES}_A(n, B', p, q)$ . (Note  $K \in \{0, \dots, q-1\}$ .)
- (1) *Modulo  $q$  guess* either one of  $q - K$  dummy accepting paths, or a list of  $B'$  configurations  $C_1, \dots, C_{B'}$  of  $\hat{M}(x)$ .

Let  $C_0 = I_n$  and  $C_{B'+1} = \hat{A}_n$ .

(2) *Modulo  $p$  guess  $i = 0, \dots, B'$ .*

If  $i < B'$  then  $s := \lceil T'(|x|)/(B' + 1) \rceil + 1$ , else  $s := T'(|x|) - B \cdot (\lceil T'(|x|)/(B' + 1) \rceil + 1)$ .

*Accept iff  $\hat{M}(x)$  run from configuration  $C_i$  for  $s$  steps does not result in  $C_{i+1}$ .*

First, note that  $B' = \Theta(B(n))$ . The runtime of Step (0) is negligible, by Lemma 6.4. We claim that Steps (1) and (2), taken as a computation with signature  $\text{MOD}_q\text{MOD}_p$ , accepts iff  $\hat{M}(x)$  accepts. Step (2) accepts if and only if the number of mistakes in the given sequence  $C_0, \dots, C_{B'+1}$  is divisible by  $p$ . Let  $t$  be the total number of complete  $B'$ -configuration sequences for  $\hat{M}(x)$  with a number of mistakes divisible by  $p$ . Then Steps (1) and (2) accept if and only if

$$q - K + t \equiv 0 \pmod{q},$$

which is equivalent to  $t \equiv K \pmod{q}$ , *i.e.*  $t \equiv \text{MISTAKES}_A(n, B, p, q) \pmod{q}$ , which is true if and only if  $\hat{M}(x)$  accepts, by Lemma 6.3.

Finally, observe that the deterministic part of the above computation (which simulates  $\hat{M}(x)$  from one configuration to the next) requires only  $O(T(n)/(B' + 1)) \leq O(T(n)/B(n))$  time and  $O(S(n))$  space. The deterministic part only takes  $x$  and two configurations as input, so its input has length  $O(n + S(n))$ .  $\square$

## 7 Lower Bound for Counting Solutions Modulo Small Integers

We now prove that there is *at most one* prime  $p$  for which  $\text{MOD}_p\text{TIME}[n] \subseteq \text{DTISP}[n^c, n^{o(1)}]$  can hold, when  $c < 1.801$ . In fact, something stronger can be shown: we shall demonstrate a “transfer principle” that says one can translate certain time-space lower bound arguments for  $\text{NTIME}[n]$ , to analogous arguments for  $\text{MOD}_p\text{TIME}[n]$ . We then prove the lower bound for  $\text{NTIME}[n]$ . Intuitively, the statement of the transfer principle says that if efficient SAT algorithms imply a contradiction with the  $\Sigma_2$  time hierarchy, then efficient  $\text{MOD}_p$ -SAT and  $\text{MOD}_q$ -SAT algorithms lead to an analogous relation for  $\text{MOD}_p\text{MOD}_q$  time.

**Transfer Principle for Time-Space Lower Bounds on MOD  $p$ :** *Suppose there is a proof of*

$$\text{NTIME}[n] \subseteq \text{DTS}[n^c] \implies \Sigma_2\text{TIME}[r(n)] \subseteq \Sigma_2\text{TIME}[s(n)]$$

*for some constant  $c$ , and polynomials  $r, s$ , where the proof uses only the following properties of nondeterminism and conondeterminism:*

1. **(Quantifier Speedup)** *For all  $B \geq 1$ ,*

$$\begin{aligned} \text{DTS}[T] &\subseteq (\exists B \cdot n^{o(1)})(\forall \log T)\text{DTS}[T/B] \\ \text{DTS}[T] &\subseteq (\forall B \cdot n^{o(1)})(\exists \log T)\text{DTS}[T/B], \end{aligned} \quad (\text{Theorem 6.1})$$

*where the inner DTS predicate of the alternating simulation reads only  $n^{o(1)}$  bits of the first quantifier.*

2. **(Quantifier Removal)** For all  $b \geq a \geq 1$ ,

$$\text{NTIME}[n] \subseteq \text{DTS}[n^c] \implies \begin{array}{l} (\exists n^a)(\forall n^b)\text{DTS}[n^b] \subseteq (\exists n^a)\text{DTS}[n^{bc}] \\ (\forall n^a)(\exists n^b)\text{DTS}[n^b] \subseteq (\forall n^a)\text{DTS}[n^{bc}] \end{array} .$$

3. **(Quantifier Combination)** For all  $a, b \geq 0$  and  $d \geq 1$ ,

$$\begin{array}{l} (\exists n^a)(\exists n^b)\text{DTS}[n^d] \subseteq (\exists n^a + n^b)\text{DTS}[n^d] \\ (\forall n^a)(\forall n^b)\text{DTS}[n^d] \subseteq (\forall n^a + n^b)\text{DTS}[n^d]. \end{array}$$

Then for all primes  $p, q$  with  $p \neq q$  and for all  $\varepsilon > 0$ ,  $\text{MOD}_p\text{TIME}[n] \subseteq \text{DTS}[n^{c-\varepsilon}]$  and  $\text{MOD}_q\text{TIME}[n] \subseteq \text{DTS}[n^{c-\varepsilon}]$  implies that there is a polynomial  $s'_\varepsilon$  such that  $\lim_{\varepsilon \rightarrow 0} s'_\varepsilon(x) = s(x)$  and

$$\text{MOD}_p\text{MOD}_q\text{TIME}[r(n)] \subseteq \text{MOD}_p\text{MOD}_q\text{TIME}[s'_\varepsilon(n)].$$

**Proof Sketch of Transfer Principle.** First, observe we have analogues to all the three properties of nondeterminism:

1. (MOD Speedup) Theorem 1.1 says that

$$\begin{array}{l} \text{DTS}[T] \subseteq (\text{MOD}_p B \cdot n^{o(1)})(\text{MOD}_q \log T)\text{DTS}[T^{1+\varepsilon}/B] \\ \text{DTS}[T] \subseteq (\text{MOD}_q B \cdot n^{o(1)})(\text{MOD}_p \log T)\text{DTS}[T^{1+\varepsilon}/B]. \end{array}$$

Moreover, the modular counting algorithm that simulates DTS reads only  $n^{o(1)}$  bits guessed in the first MOD mode of the algorithm.

2. (MOD Removal) If  $\text{MOD}_p\text{TIME}[n] \subseteq \text{DTS}[n^c]$  and  $\text{MOD}_q\text{TIME}[n] \subseteq \text{DTS}[n^c]$ , then for  $b \geq a \geq 1$ ,

$$\begin{array}{l} (\text{MOD}_p n^a)(\text{MOD}_q n^b)\text{DTS}[n^b] \subseteq (\text{MOD}_p n^a)\text{DTS}[n^{ac}] \\ (\text{MOD}_q n^a)(\text{MOD}_p n^b)\text{DTS}[n^b] \subseteq (\text{MOD}_q n^a)\text{DTS}[n^{bc}], \end{array}$$

by padding.

3. (MOD Combination) For all primes  $p$ ,  $(\text{MOD}_p n^a)(\text{MOD}_p n^b)\text{DTS}[n^d] \subseteq (\text{MOD}_p n^a + n^b)\text{DTS}[n^d]$ . (This was shown by Beigel and Gill [BG92]– the proof uses Fermat's Little Theorem that  $a^{p-1} \equiv 1 \pmod{p}$ .)

Let  $\varepsilon > 0$ . By assumption, a proof that  $\text{NTIME}[n] \subseteq \text{DTS}[n^c]$  implies  $\Sigma_2\text{TIME}[r(n)] \subseteq \Sigma_2\text{TIME}[s(n)]$  starts with the class  $\Sigma_2\text{TIME}[r(n)]$  and uses the three properties in various ways to derive new containments, until the class  $\Sigma_2\text{TIME}[s(n)]$  is reached. To prove that  $\text{MOD}_p\text{TIME}[n] \subseteq \text{DTS}[n^{c-\varepsilon}]$  and  $\text{MOD}_q\text{TIME}[n] \subseteq \text{DTS}[n^{c-\varepsilon}]$  imply  $\text{MOD}_p\text{MOD}_q\text{TIME}[r(n)] \subseteq \text{MOD}_p\text{MOD}_q\text{TIME}[s'_\varepsilon(n)]$ , we start with the class  $\text{MOD}_p\text{MOD}_q\text{TIME}[r(n)]$  and use the MOD analogues of the three properties to derive similar containments. In particular, every  $\exists$  quantifier ( $\forall$  quantifier) in the original proof is systematically replaced with a  $\text{MOD}_p$  mode (respectively,  $\text{MOD}_q$  mode) in the proof of  $\text{MOD}_p\text{MOD}_q\text{TIME}[r(n)] \subseteq \text{MOD}_p\text{MOD}_q\text{TIME}[s'_\varepsilon(n)]$ .

The Quantifier Removal and Quantifier Combination properties have exactly the same parameters as the MOD Removal and MOD Combination properties, but each invocation of MOD Speedup

(instead of Quantifier Speedup) in the new proof results in an extra  $\varepsilon$  factor in the exponent. However, the resulting polynomial runtime  $s'_\varepsilon(n)$  of the final  $\text{MOD}_p\text{MOD}_q$  class still has the property that if  $\varepsilon = 0$ , then the derived polynomial is exactly  $s(n)$ . So  $\lim_{\varepsilon \rightarrow 0} s'_\varepsilon(n) = s(n)$ .  $\square$

The transfer principle implies two significant corollaries. Both of them say that certain proofs-by-contradiction that  $\text{NTIME}[n] \not\subseteq \text{DTS}[n^c]$  can be mapped over to proofs that  $\text{MOD}_k\text{TIME}[n] \not\subseteq \text{DTS}[n^{c-\varepsilon}]$ , for various integers  $k$ .

**Corollary 7.1** *Suppose there is a proof of  $\text{NTIME}[n] \subseteq \text{DTS}[n^c] \implies \Sigma_2\text{TIME}[n^k] \subseteq \Sigma_2\text{TIME}[n^{k-\delta}]$  (for some  $k > 1$  and  $\delta > 0$ ) that follows the three conditions of the transfer principle. Then for every  $m > 1$  that is not a prime power,  $\text{MOD}_m\text{TIME}[n] \not\subseteq \text{DTS}[n^{c-\varepsilon}]$  for all  $\varepsilon > 0$ .*

If  $\text{NTIME}[n] \subseteq \text{DTS}[n^c] \implies \Sigma_2\text{TIME}[n^k] \subseteq \Sigma_2\text{TIME}[n^{k-\delta}]$ , then  $\text{NTIME}[n] \not\subseteq \text{DTS}[n^c]$ , by the following simple separation.

**Theorem 7.1 (Folklore)**  $\Sigma_2\text{TIME}[t(n)] \not\subseteq \Sigma_2\text{TIME}[t(n)^{1-\varepsilon}]$  for all time constructible  $t(n)$  and  $\varepsilon > 0$ .

There is a modular analogue of this separation as well, which is straightforward.

**Theorem 7.2**  $\text{MOD}_p\text{MOD}_q\text{TIME}[t(n)] \not\subseteq \text{MOD}_p\text{MOD}_q\text{TIME}[t(n)^{1-\varepsilon}]$  for all time constructible  $t(n)$  and  $\varepsilon > 0$ .

**Proof of Corollary 7.1.** If  $m$  is not a prime power, then it has two distinct primes  $p$  and  $q$  as factors. Therefore,  $\text{MOD}_p\text{TIME}[t] \subseteq \text{MOD}_m\text{TIME}[t]$  and  $\text{MOD}_q\text{TIME}[t] \subseteq \text{MOD}_m\text{TIME}[t]$  for all time bounds  $t$  (cf. [BG92] for a proof). Thus  $\text{MOD}_m\text{TIME}[n] \subseteq \text{DTS}[n^{c-\varepsilon}]$  implies

$$\text{MOD}_p\text{TIME}[n] \subseteq \text{DTS}[n^{c-\varepsilon}] \quad \text{and}$$

$$\text{MOD}_q\text{TIME}[n] \subseteq \text{DTS}[n^{c-\varepsilon}].$$

Therefore, one can use the same proof used by the transfer principle to obtain the inclusion  $\text{MOD}_p\text{MOD}_q\text{TIME}[n^k] \subseteq \text{MOD}_p\text{MOD}_q\text{TIME}[s'_\varepsilon(n)]$ , where  $s'_\varepsilon(n)$  is a polynomial that converges to  $n^{k-\delta}$  for  $\varepsilon \rightarrow 0$ , contradicting Theorem 7.2.  $\square$

**Corollary 7.2** *Suppose there is a proof of  $\text{NTIME}[n] \subseteq \text{DTS}[n^c] \implies \Sigma_2\text{TIME}[n^k] \subseteq \Sigma_2\text{TIME}[n^{k-\delta}]$  (for some  $k > 1$  and  $\delta > 0$ ) that follows the three conditions of the transfer principle. Then, for every prime  $p$  except for possibly one of them,  $\text{MOD}_p\text{TIME}[n] \not\subseteq \text{DTS}[n^{c-\varepsilon}]$  for all  $\varepsilon > 0$ . That is, there is at most one prime  $p$  for which  $\text{MOD}_p\text{TIME}[n]$  is in  $n^{c-\varepsilon}$  time and  $n^{o(1)}$  space.*

**Proof.** If there are two primes  $p$  and  $q$  such that  $\text{MOD}_p\text{TIME}[n] \subseteq \text{DTS}[n^{c-\varepsilon}]$  and  $\text{MOD}_q\text{TIME}[n] \subseteq \text{DTS}[n^{c-\varepsilon}]$ , then the transfer principle applies as in the previous corollary, and we can establish a contradiction. So at most one prime  $p$  can satisfy  $\text{MOD}_p\text{TIME}[n] \subseteq \text{DTS}[n^{c-\varepsilon}]$ .  $\square$

We now proceed with a new time-space lower bound for  $\text{NTIME}[n]$ . Theorem 1.2 follows immediately from it, coupled with the above two results.

**Theorem 7.3** For all  $c \geq 1$  such that  $c^3 - c^2 - 2c + 1 < 0$ ,

$$\text{NTIME}[n] \not\subseteq \text{DTS}[n^c].$$

The proof proceeds by showing  $\text{NTIME}[n] \subseteq \text{DTS}[n^c]$  implies  $\Sigma_2\text{TIME}[n^k] \subseteq \Sigma_2\text{TIME}[n^{k-\delta}]$  for some  $k \geq 1$  and  $\delta > 0$ . Furthermore, the proof uses only the three properties of nondeterminism required in the transfer principle.

**Proof.** See Appendix. □

## 8 Deterministic Small Space Versus Modular Counting

Using the Speedup By Modular Counting Theorem, one can prove modular-counting analogues of known results about the power of alternating computation. For example, if  $\text{MOD}_6\text{TIME}[n^k]$  is efficiently closed under Turing reductions for any  $k$ , then a major separation result would be obtained. Recall that  $\text{SC} := \text{DTISP}[n^{O(1)}, (\log n)^{O(1)}]$ .

**Theorem 8.1** If there is some  $k \geq 1$  such that  $(\text{MOD}_6\text{MOD}_6)\text{TIME}[n^k] \subseteq \text{MOD}_6\text{TIME}[n^{k+o(1)}]$ , then  $\text{SC} \neq \text{MOD}_6\text{P}$ .

That is, if a  $\text{MOD}_6$  machine that makes oracle calls to another  $\text{MOD}_6$  machine can be efficiently simulated by a single  $\text{MOD}_6$  machine, then it follows that logspace is different from  $\text{MOD}_6\text{P}$ . This theorem is an analogue of the following result for nondeterminism which follows from the work of Fortnow [For97].

**Theorem 8.2** If  $\text{NTIME}[n^k] \subseteq \text{coNTIME}[n^{k+o(1)}]$  then  $\text{L} \neq \text{NP}$ .

**Proof of Theorem 8.1.** It follows from the Speedup by Modular Counting Theorem that for any integer  $\ell \geq 1$  and real  $k \geq 1$ ,  $\text{DTISP}[n^{\ell+k}, (\log n)^{\ell+k}]$  can be simulated by a random access machine of type  $(\text{MOD}_6)^\ell$  (i.e.  $k$  separate  $\text{MOD}_6$  modes) that guesses  $O(n)$  bits in each mode, and the last deterministic part runs in  $n^{k+\varepsilon}$  time, for any  $\varepsilon > 0$ . By the assumption, the first  $\ell - 1$   $\text{MOD}_6$  modes of this machine can be “removed”, while only increasing the runtime by a  $o(1)$  factor in the exponent. Therefore

$$\text{DTISP}[n^{k+\ell}, (\log n)^{2k}] \subseteq \text{MOD}_6\text{TIME}[n^{k+o(1)}] \quad (*)$$

for any  $\ell \geq 1$ . But then if  $\text{MOD}_6\text{TIME}[n] \subseteq \text{DTISP}[n^j, (\log n)^j]$  for some  $j$ ,

$$\text{MOD}_6\text{TIME}[n^{1+k}] \subseteq \text{DTISP}[n^{j(1+k)}, (\log n)^{j(1+k)}] \subseteq \text{MOD}_6\text{TIME}[n^{k+o(1)}],$$

by setting  $\ell = (j(1+k) - k)$  and applying (\*). This is a contradiction to the  $\text{MOD}_6$  time hierarchy. Therefore  $\text{SC} \neq \text{MOD}_6\text{P}$ . □

Similarly, the following can be proved.

**Theorem 8.3** If there exist primes  $p$  and  $q$  and  $k \geq 1$  such that  $\text{MOD}_p\text{TIME}[n^k] \subseteq \text{MOD}_q\text{TIME}[n^{k+o(1)}]$  and  $\text{MOD}_q\text{TIME}[n^k] \subseteq \text{MOD}_p\text{TIME}[n^{k+o(1)}]$ , then  $\text{SC} \neq \text{MOD}_p\text{P}$  and  $\text{SC} \neq \text{MOD}_q\text{P}$ .

Finally, we give an unconditional separation. Fortnow [For97] showed that NL is not equal to any non-constant level of the polynomial time hierarchy. We can give an analogous separation result for the class SC. For an integer  $m$ , define a polynomial time MOD $_m$ -hierarchy by

$$\begin{aligned}(\text{MOD}_m)^0\text{P} &:= \text{P} \\ (\text{MOD}_m)^k\text{P} &:= (\text{MOD}_m)[(\text{MOD}_m)^{k-1}\text{P}].\end{aligned}$$

**Theorem 8.4** *Let  $s(n)$  be any monotone increasing unbounded function. Then  $\text{SC} \neq (\text{MOD}_m)^{s(n)}\text{P}$ , for any composite  $m$  that is not a prime power.*

## 9 Conclusion

We have proven the first superlinear time lower bounds for counting NP solutions modulo small integers, on random access machines that use subpolynomial space. For example, the best known time-space lower bound for MOD $_6$ -SAT is the same as that for SAT. To arrive at our results, we discovered a way to transfer lower bound proofs for nondeterminism to the setting of modular counting, using a lemma that shows the number of configuration sequences of a canonical machine with a prescribed number of mistakes on a given input depends solely upon the space usage of the machine and the acceptance/rejection condition of that input. Such a tool may prove to be useful in other time-space lower bound arguments as well. For example, one may be able to prove strong time-space lower bounds for the MAJORITY SAT problem using the Counting Lemma, but we have not yet found a way to do this. Currently the best time-space lower bound we know for MAJORITY-SAT is the same as the one known for SAT.

A time-space lower bound for MOD $_p$ -SAT for a particular prime  $p$  follows from the following conjectured time hierarchy, which we consider to be an interesting open problem.

**Conjecture:** There are primes  $q \neq p$ , and some  $t(n) > n$ , such that  $\text{MOD}_p\text{TIME}[t] \not\subseteq \text{MOD}_q\text{TIME}[o(t)]$ .

We also gave a new argument that improves the time lower bound for SAT on subpolynomial space RAMs, to  $\Omega(n^{1.801})$ . We remain confident that a proof of a quadratic (or larger) lower bound should be possible, by extending our techniques.

## 10 Acknowledgements

I am grateful to Virginia Vassilevska, Dieter van Melkebeek, and the anonymous referees for their helpful comments.

## References

- [AG94] E. Allender and V. Gore. A uniform circuit lower bound for the permanent. *SIAM J. Comput.* 23:1026–1049, 1994.

- [All98] E. Allender. The Permanent Requires Large Uniform Threshold Circuits. *Chicago Journal of Theoretical Computer Science*, article 7, 1999.
- [AKRRV01] E. Allender, M. Koucký, D. Ronneburger, S. Roy, and V. Vinay. Time-Space Tradeoffs in the Counting Hierarchy. In *IEEE Conference on Computational Complexity*, 295–302, 2001.
- [BST90] D. Barrington, H. Straubing, and D. Therien. Non-uniform automata over groups. *Information and Computation* 89(2):109–132, 1990.
- [BBF98] R. Beigel, H. Buhrman, and L. Fortnow. NP might not be as easy as detecting unique solutions. In *Proc. ACM STOC*, 203–208, 1998.
- [BG92] R. Beigel and J. Gill. Counting classes: thresholds, parity, mods, and fewness. *Theoretical Computer Science* 103(1):3–23, 1992.
- [Ben89] C. H. Bennett. Time-space tradeoffs for reversible computation. *SIAM J. Comput.* 18:766–776, 1989.
- [BTV01] H. Buhrman, J. Tromp, and P. Vitányi. Time and Space Bounds for Reversible Simulation. *J. Phys. A: Math. Gen.* 34:6821–6830, 2001.
- [CH89] J.-Y. Cai and L. A. Hemachandra. On the power of parity polynomial time. In *Proc. Symposium on Theoretical Aspects of Computer Science (STACS)*, Springer LNCS 349, 229–240, 1989.
- [CC06] J.-Y. Cai and V. Choudhary. Some Results on Matchgates and Holographic Algorithms. In *Proc. of ICALP Vol. 1*, Springer-Verlag LNCS, 703–714, 2006.
- [CP07] J.-Y. Cai and P. Lu. Bases Collapse in Holographic Algorithms. To appear in *Proc. IEEE Conference on Computational Complexity*, 2007.
- [CMTV98] H. Caussinus, P. McKenzie, D. Therien and H. Vollmer. Nondeterministic NC1 Computation. *J. Comput. Syst. Sci.* 57(2):200–212, 1998.
- [CKS81] A. K. Chandra, D. Kozen, and L. J. Stockmeyer. Alternation. *JACM* 28(1):114–133, 1981.
- [CGPT06] A. Chattopadhyay, N. Goyal, P. Pudlak, D. Therien. Lower bounds for circuits with  $\text{MOD}_m$  gates. In *Proc. IEEE FOCS*, 2006.
- [DvM06] S. Diehl and D. van Melkebeek. Time-Space Lower Bounds for the Polynomial-Time Hierarchy on Randomized Machines. *SIAM J. Comput.* 36: 563-594, 2006.
- [For97] L. Fortnow. Nondeterministic Polynomial Time Versus Nondeterministic Logarithmic Space: Time-Space Tradeoffs for Satisfiability. In *Proc. IEEE Conference on Computational Complexity*, 52–60, 1997.
- [FLvMV05] L. Fortnow, R. Lipton, D. Van Melkebeek, and A. Viglas. Time-Space Lower Bounds for Satisfiability. *JACM* 52(6):835–865, 2005.

- [FvM00] L. Fortnow and D. van Melkebeek. Time-Space Tradeoffs for Nondeterministic Computation. In *Proc. IEEE Conference on Computational Complexity*, 2–13, 2000.
- [Gro01] V. Grolmusz. A Degree-Decreasing Lemma for (MOD-q - MOD-p) Circuits. *Disc. Math. and Theor. Comp. Sci.* 4(2):247–254, 2001.
- [Gup98] S. Gupta. Isolating an Odd Number of Elements and Applications in Complexity Theory. *Theory of Computing Systems* 31:27–40, 1998.
- [Kan83] R. Kannan. Alternation and the power of nondeterminism. In *Proceedings of ACM STOC*, 344–346, 1983.
- [Kan84] R. Kannan. Towards Separating Nondeterminism from Determinism. *Mathematical Systems Theory* 17(1):29–45, 1984.
- [KvM02] A. Klivans and D. Van Melkebeek. Graph Nonisomorphism Has Subexponential Size Proofs Unless the Polynomial-Time Hierarchy Collapses. *SIAM J. Comput.* 31:1501–1526, 2002.
- [LV99] R. J. Lipton and A. Viglas. On the Complexity of SAT. In *Proceedings of IEEE FOCS*, 459–464, 1999.
- [MS87] W. Maass and A. Schorr. Speed-Up of Turing Machines with One Work Tape and a Two-Way Input Tape. *SIAM J. Comput.* 16(1):195–202, 1987.
- [NRS95] A. V. Naik, K. W. Regan, D. Sivakumar. On quasilinear-time complexity theory. *Theoretical Computer Science* 148:325–349, 1995.
- [Nep70] V. Nepomnjascii. Rudimentary predicates and Turing calculations. *Soviet Math. Doklady* 11:1462–1465, 1970.
- [TO92] S. Toda and M. Ogiwara. Counting Classes are at Least as Hard as the Polynomial-Time Hierarchy. *SIAM J. Comput.* 21(2):316–328, 1992.
- [PPST83] W. Paul, N. Pippenger, E. Szemerédi, and W. Trotter. On determinism versus nondeterminism and related problems. In *Proc. IEEE FOCS*, 429–438, 1983.
- [Smo87] R. Smolensky. Algebraic methods in the theory of lower bounds for Boolean circuit complexity. In *Proc. ACM STOC*, 77–82, 1987.
- [Tou01] I. Tourlakis. Time-Space Tradeoffs for SAT on Nonuniform Machines. *J. Computer and System Sciences* 63(2): 268–287, 2001.
- [VV86] L. Valiant and V. Vazirani. NP is as easy as detecting unique solutions. *TCS* 47(1):85–93, 1986.
- [Val04] L. Valiant. Holographic algorithms. In *Proc. IEEE FOCS*, 306–315, 2004.
- [Val06] L. Valiant. Accidental algorithms. In *Proc. IEEE FOCS*, 509–517, 2006.
- [Wil05] R. Williams. Better Time-Space Lower Bounds for SAT and Related Problems. In *Proc. IEEE Conference on Computational Complexity*, 40–49, 2005.

## 11 Appendix

Recall the statement we wish to prove:

**Theorem 7.3:** *For all  $c \geq 1$  such that  $c^3 - c^2 - 2c + 1 < 0$ ,*

$$\text{NTIME}[n] \not\subseteq \text{DTS}[n^c].$$

*Furthermore, the proof uses only the three properties of nondeterminism required in the transfer principle.*

We first need a speedup simulation from our previous work [Wil05]. Effectively, this simulation says that if we assume that nondeterministic linear time can be simulated in  $n^c$  time and  $n^{o(1)}$  space, that assumption can be applied to improve the special case of Nepomnjascii's theorem, proven earlier in Theorem 6.1.

**Theorem 11.1 (Conditional Speedup [Wil05])** *Suppose there is a  $c < 2$  satisfying  $\text{NTIME}[n] \subseteq \text{DTS}[n^c]$ . Then for all  $d < c/(c-1)$ ,*

$$\text{DTS}[n^d] \subseteq (\exists n^{1+o(1)})(\forall \log n)\text{DTS}[n^{1+o(1)}] \cap (\forall n^{1+o(1)})(\exists \log n)\text{DTS}[n^{1+o(1)}].$$

*Moreover, the proof applies only the three properties required by the transfer principle.*

Notice that when  $c < 2$ , we have that  $c/(c-1) > 2$ . Therefore, the alternating speedup of  $n^d$  time and  $n^{o(1)}$  obtained is *better* than the square root gotten from Theorem 6.1.

**Proof.** Define the sequence  $d_0 := 2$ ,  $d_k := 1 + \frac{d_{k-1}}{c}$ . We prove by induction that for all  $k \geq 0$ ,

$$\text{DTS}[n^{d_k}] \subseteq (\exists n^{1+o(1)})(\forall \log n)\text{DTS}[n^{1+o(1)}] \cap (\forall n^{1+o(1)})(\exists \log n)\text{DTS}[n^{1+o(1)}].$$

As the sequence  $\{d_k\}$  is monotone increasing and converges to  $c/(c-1)$  (when  $c < 2$ ), the theorem follows.

The case  $k = 0$  follows from Theorem 6.1, as it says  $\text{DTS}[n^2] \subseteq \Sigma_2 \text{TIME}[n^{1+o(1)}] \cap \Pi_2 \text{TIME}[n^{1+o(1)}]$ , using property 1 (Quantifier Speedup) of the transfer principle.

For the inductive step, consider the class  $\text{DTS}[n^{d_{k+1}}] = \text{DTS}[n^{1+d_k/c}]$ . By property 1 (Quantifier Speedup),  $\text{DTS}[n^{1+d_k/c}]$  is contained in

$$(\exists n)(\forall \log n)\text{DTS}[n^{d_k/c+o(1)}].$$

Note that  $d_k/c \geq 1$ . The conondeterministic part of the  $\Sigma_2$  class above always has input of length  $O(n)$ . Therefore by property 2 (Quantifier Removal) of the transfer principle, we can apply the assumption  $\text{NTIME}[n] \subseteq \text{DTS}[n^c]$  to the  $(\forall \log n)\text{DTS}[n^{d_k/c+o(1)}]$  part, obtaining the class

$$(\exists n)\text{DTS}[n^{d_k+o(1)}].$$

Finally, by induction hypothesis,  $\text{DTS}[n^{d_k+o(1)}]$  is contained in  $\Sigma_2 \text{TIME}[n^{1+o(1)}]$ , so the above class is contained in

$$(\exists n)(\exists n^{1+o(1)})(\forall \log n)\text{DTS}[n^{1+o(1)}] \subseteq (\exists n^{1+o(1)})(\forall \log n)\text{DTS}[n^{1+o(1)}],$$

by property 3 (Quantifier Combination) of the transfer principle.

An similar argument shows that  $\text{DTS}[n^{d_{k+1}}] \subseteq (\forall n^{1+o(1)})(\exists \log n)\text{DTS}[n^{1+o(1)}]$  as well.  $\square$

The bulk of the proof is in the following result, which is a subtle combination of the proof strategy of Fortnow and Van Melkebeek and the Conditional Speedup Theorem 11.1.

**Theorem 11.2** *Suppose there is  $c < 2$  such that  $\text{NTIME}[n] \subseteq \text{DTS}[n^c]$ . Then for all integers  $k \geq 1$ , and  $d < c/(c-1)$ ,*

$$\text{DTS}[n^{d+\sum_{i=1}^k (c^2/d)^i}] \subseteq \Sigma_2 \text{TIME}[n^{c^2(d+\sum_{i=1}^k (c^2/d)^i)+o(1)}] \cap \Pi_2 \text{TIME}[n^{c^2(d+\sum_{i=1}^k (c^2/d)^i)+o(1)}].$$

*The proof applies only the three properties required by the transfer principle.*

We first show how Theorem 11.2 implies Theorem 7.3, the  $n^{1.8}$  lower bound.

**Proof of Theorem 7.3.** Assuming  $\text{NTIME}[n] \subseteq \text{DTS}[n^c]$  and Theorem 11.2,

$$\Sigma_2 \text{TIME}[n^{d+\sum_{i=1}^k (c^2/d)^i}] \subseteq \text{NTIME}[n^{c(d+\sum_{i=1}^k (c^2/d)^i)}] \subseteq \text{DTS}[n^{c^2(d+\sum_{i=1}^k (c^2/d)^i)}] \subseteq \Sigma_2 \text{TIME}[n^{c^2(d+\sum_{i=1}^k (c^2/d)^i)+o(1)}].$$

A contradiction with the  $\Sigma_2$  time hierarchy (Theorem 7.1) is reached (and therefore  $\text{NTIME}[n] \not\subseteq \text{DTS}[n^c]$ ) precisely when

$$d + \sum_{i=1}^k (c^2/d)^i > c^2 \cdot (c^2/d)^k,$$

that is, when

$$\begin{aligned} c^2 < \sum_{i=1}^k \left(\frac{c^2}{d}\right)^{i-k} + d \cdot \frac{d^k}{c^{2k}} &\iff c^2 < \sum_{j=0}^{k-1} \left(\frac{d}{c^2}\right)^j + d \cdot \left(\frac{d}{c^2}\right)^k \\ &\iff c^2 < \frac{1 - \left(\frac{d}{c^2}\right)^k}{1 - \left(\frac{d}{c^2}\right)} + d \cdot \left(\frac{d}{c^2}\right)^k \quad (*) \end{aligned}$$

Note that  $c^2 \geq d$ , since Fortnow *et al.* [FLVMV05] proved that  $c$  must be at least the golden ratio, therefore  $c(c-1) \geq 1$ , *i.e.*  $c^2 \geq c/(c-1) > d$ . Therefore  $1 > (d/c^2)^k$  for all  $k \geq 1$ , and

$$\lim_{k \rightarrow \infty} (d/c^2)^k = 0.$$

Hence for any  $\varepsilon > 0$ , we can set  $d = c/(c-1) - \varepsilon$  and find a  $k$  such that  $((c/(c-1) - \varepsilon)/c^2)^k \leq \varepsilon$ , whereby the inequality (\*) turns into

$$c^2 < \frac{1 - \varepsilon}{1 - \frac{c/(c-1) - \varepsilon}{c^2}} + (c/(c-1) - \varepsilon) \cdot \varepsilon.$$

Simple algebraic manipulation yields the equivalent condition:

$$c^2 - (c/(c-1) - \varepsilon) < (1 - \varepsilon) + (c/(c-1) - \varepsilon) \cdot \varepsilon \cdot \left(1 - \frac{c/(c-1) - \varepsilon}{c^2}\right)$$

Multiplying through by  $(c - 1)$ , the condition becomes

$$\begin{aligned} c^2(c - 1) - (c - \varepsilon(c - 1)) &< ((c - 1) - \varepsilon(c - 1)) + (c - \varepsilon(c - 1)) \cdot \varepsilon \cdot \left(1 - \frac{c/(c - 1) - \varepsilon}{c^2}\right) \\ \iff c^3 - c^2 - 2c + 1 &< (c - \varepsilon(c - 1)) \cdot \varepsilon \cdot \left(1 - \frac{c/(c - 1) - \varepsilon}{c^2}\right) - 2\varepsilon(c - 1) \quad (**) \end{aligned}$$

Now, as  $\varepsilon$  approaches 0, the RHS approaches 0. We arrive at the following condition implying a contradiction:

$$c^3 - c^2 - 2c + 1 < 0,$$

which is what we wanted to prove. That is, for any  $c$  satisfying  $c^3 - c^2 - 2c + 1 < 0$ , we can choose an  $\varepsilon > 0$  such that  $c$  and  $\varepsilon$  satisfy (\*\*).  $\square$

**Proof of Theorem 11.2.** Suppose  $c < 2$  is such that  $\text{NTIME}[n] \subseteq \text{DTS}[n^c]$ . Pick  $d$  such that  $c \leq d < c/(c - 1)$ . The proof is by induction on  $k$ .

For  $k = 0$ , the task is just to show  $\text{DTS}[n^d] \subseteq \Sigma_2 \text{TIME}[n^{1+o(1)}]$ , which is precisely Theorem 11.1. For the inductive step, consider the class  $\text{DTS}[n^{d+\sum_{i=1}^k (c^2/d)^k}]$ . By Theorem 6.1,

$$\text{DTS}[n^{d+\sum_{i=1}^k (c^2/d)^k}] \subseteq (\exists n^{(c^2/d)^k})(\forall \log n) \text{DTS}[n^{d+\sum_{i=1}^k (c^2/d)^{k-1}}],$$

where the  $\text{DTS}[\dots]$  part of the  $\Sigma_2$  computation has input of length  $n + n^{o(1)}$  (the original input  $x$ , and two configurations). This step uses property 1 (Quantifier Speedup) of the transfer principle. By the induction hypothesis,

$$(\exists n^{(c^2/d)^k})(\forall \log n) \text{DTS}[n^{d+\sum_{i=1}^k (c^2/d)^{k-1}}] \subseteq (\exists n^{(c^2/d)^k})(\forall \log n) \Pi_2 \text{TIME}[n^{(c^2/d)^{k-1}+o(1)}].$$

Applying the assumption that  $\text{NTIME}[n] \subseteq \text{DTS}[n^c]$  via property 2 (Quantifier Removal) to the  $\Pi_2$  part (which takes input of length  $n + n^{o(1)}$ ),

$$(\exists n^{(c^2/d)^k})(\forall \log n) \Pi_2 \text{TIME}[n^{(c^2/d)^{k-1}+o(1)}] \subseteq (\exists n^{(c^2/d)^k})(\forall \log n) \text{coNTIME}[n^{c \cdot (c^2/d)^{k-1}+o(1)}].$$

By property 3 (Quantifier Combination),

$$(\exists n^{(c^2/d)^k})(\forall \log n) \text{coNTIME}[n^{c \cdot (c^2/d)^{k-1}+o(1)}] \subseteq (\exists n^{(c^2/d)^k}) \text{coNTIME}[n^{c \cdot (c^2/d)^{k-1}+o(1)}].$$

Now the input to the  $\text{coNTIME}$  part is  $O(n^{(c^2/d)^k}) \leq O(n^{c \cdot (c^2/d)^{k-1}})$ , since  $d \geq c$ . Therefore we can use Quantifier Removal again to obtain

$$(\exists n^{(c^2/d)^k}) \text{coNTIME}[n^{c \cdot (c^2/d)^{k-1}+o(1)}] \subseteq (\exists n^{(c^2/d)^k}) \text{DTS}[n^{c^2 \cdot (c^2/d)^{k-1}+o(1)}].$$

But by Theorem 11.1, the  $\text{DTS}$  part of the above can be replaced with a  $\Sigma_2$  computation, in particular

$$(\exists n^{(c^2/d)^k}) \text{DTS}[n^{c^2 \cdot (c^2/d)^{k-1}+o(1)}] \subseteq (\exists n^{(c^2/d)^k})(\exists n^{(c^2/d)^k+o(1)})(\forall \log n) \text{DTS}[n^{(c^2/d)^k+o(1)}].$$

Finally,

$$(\exists n^{(c^2/d)^k})(\exists n^{(c^2/d)^k+o(1)})(\forall \log n) \text{DTS}[n^{(c^2/d)^k+o(1)}] \subseteq (\exists n^{(c^2/d)^k+o(1)})(\forall \log n) \text{DTS}[n^{(c^2/d)^k+o(1)}]$$

by Quantifier Combination. This completes the proof.  $\square$