Institute for Language, Logic and Information

HONEST REDUCTIONS, COMPLETENESS AND NONDETERMINISTIC COMPLEXITY CLASSES

Harry Buhrman
Steven Homer
Leen Torenvliet

ITLI Prepublication Series
for Computation and Complexity Theory CT-89-08

University of Amsterdam

Instituut voor Taal, Logica en Informatie
Institute for Language, Logic and Information

Faculteit der Wiskunde en Informatica
(Department of Mathematics and Computer Science)
Plantage Muidergracht 24
1018TV Amsterdam

Faculteit der Wijsbegeerte
(Department of Philosophy)
Nieuwe Doelenstraat 15
1012CP Amsterdam

HONEST REDUCTIONS, COMPLETENESS AND NONDETERMINISTIC COMPLEXITY CLASSES

Harry Buhrman
Department of Mathematics
and Computer Science
University of Amsterdam

Steven Homer
Computer Science Department
Boston University
Boston, MA 02215

Leen Torenvliet
Department of Mathematics
and Computer Science
University of Amsterdam

Steven Homer was supported in part by
National Science Foundation Grants
MIP-8608137 and CCR-8814339
and a Fulbright-Hays Research Fellowship.
Some of this research was done while the
author was a Guest Professor at the
Mathematics Institute of
Heidelberg University

Received November 1989

Abstract

We demonstrate differences between reducibilities and corresponding completeness notions for nondeterministic time- and space classes. For time classes the studied completeness notions under polynomial-time bounded (even logarithmic space bounded) reducibilities turn out to be different for any class containing *NE*. For space classes the completeness notions under logspace reducibilities can be separated for any class properly containing *LOGSPACE*. Key observation in obtaining the separations is the honesty property of reductions, which was recently observed to hold for the time classes and can be shown to hold for space classes. For the case of truth-table reductions we give a new interpretation of this notion.

1 Introduction

Efficient reducibilities and completeness are two of the central concepts of complexity theory. Since the first use of polynomial time bounded Turing reductions by Cook [4] and the introduction of polynomial time bounded many-one reductions by Karp[9], considerable effort has been put in the investigation of properties and the relative strengths of different reductions and corresponding completeness notions. In 1975 Landner, Lynch and Selman [11] gave an extensive survey of different types of reductions and differences between these reductions on $E\,(=\cup_{c\in\mathbb{N}}\mathrm{DTIME}\,(2^{cn}))$. However, they did not present any conclusions concerning any differences in complete sets for these various reductions. In particular they left open the question of whether these different reductions yield different complete sets. In 1987, Watanabe [13] building upon earlier work of L. Berman [2], proved almost all possible differences between the polynomial-time completeness notions on E and larger deterministic time classes.

The question of differentiating between complete sets for nondeterministic time classes with respect to the various bounded reductions is left as an open problem in Watenabe [13]. In section 3 we solve this problem and prove the separation of complete sets for the nondeterministic exponential time class $NE\,(=\cup_{c\in\mathbb{N}}\mathrm{NTIME}\,(2^{cn}))$ with respect to the most important types of polynomial time reductions: many-one, bounded truth table, truth table and Turing. The results hold for many larger non-deterministic classes as well. The main new tool here is the work of Ganesan and Homer [5] on the structure of complete sets for nondeterministic classes and in particular a careful examination of the honesty of the various reductions. Further, by generalizing the results in [5] we are also able to separate the corresponding completeness notions arising from logspace reductions.

Logspace reductions have long been studied as strengthenings of polynomial-time reductions. It was realized early-on that many NP-complete sets (with respect to many one polynomial-time reductions) were in fact log-space complete for NP [9,10]. Shortly after the initial work on the isomorphism conjecture by Berman and Hartmanis [3], Hartmanis [7] studied the question of the logspace isomorphism of NP-complete sets and achieved many of the same results as in [3] for logspace reductions.

An important advantage of logspace reductions is that they give rise to the definition and study of complete sets for smaller complexity classes such as NL, CSL and P. The properties of such complete sets have consequences for parallel algorithms and the study of parallel complexity classes. Nonetheless, logspace reductions have been much less studied than have polynomial-time reductions. In Section 4 we consider the problem of differentiating the various completeness notions in nondeterministic space bounded classes for the different types of logspace reductions. We prove that logspace many-one, disjunctive, bounded truth-table and truth-table completeness differ on nondeterministic space classes bigger than logspace. The methods used are similar to those of Watanabe [13], but the key new idea is an application of the recent theorem of Immerman [8] and (independently) of Szelepscényi [12] showing closure of these classes under complement.

2 Preliminaries

2.1 Machines and languages

Let $\Sigma = \{0, 1\}$. Strings are elements of Σ^*, and are denoted by small letters $x, y, u, v, \ldots$. For any string x the length of a string is denoted by $|x|$. Languages are subsets of Σ^*, and are denoted by capital letters $A, B, C, S, \ldots$. For any set S the cardinality of S is denoted by $|S|$. For any set A the set $A^{\leq n}$ consists of all strings in A of length $\leq n$. We fix a pairing function $\lambda xy.<x, y>$ computable in logarithmic space and polynomial time from $\Sigma^* \times \Sigma^*$ to Σ^*. We will use the following notation:

$\overset{\infty}{\exists}\, n$: for infinitely many n

$\overset{\infty}{\forall}\, n$: for all but finitely many n

We assume that the reader is familiar with the standard Turing machine model. An *oracle* machine is a multi-tape Turing machine with an input tape, an output tape, worktapes, and a *query* tape. Oracle machines have three distinguished states QUERY, YES and NO, which are explained as follows: at some stage(s) in the computation the machine may enter the state QUERY and then goes to the state YES or goes to the state NO depending on the membership of the string currently written on the query tape in a fixed *oracle* set.

Oracle machines appear in the paper in two flavors: adaptive and non-adaptive. For a non-adaptive machine queries may not be interdependent, whereas an adaptive machine may compute a next query depending on the answer to previous queries.

Whenever it is obvious that a universal recognizing or transducing machine exists for a class of languages (i.e. the class is recursively presentable), we will assume an enumeration of the acceptors and/or transducers and denote this enumeration by $M_1, M_2, \ldots$ and $f_1, f_2, \ldots$. Examples are:

- The class of polynomial time and/or logarithmic space bounded transducers.
- The class of polynomial time bounded constant-query bounded oracle machines.

We use $M^A(x)$ to denote the computation of M on input x relative to oracle A. Let the set of queries made by M^A during this computation be denoted by $Q(M, x, A)$ if M is an adaptive machine, and by $Q(M, x)$ if M is a non-adaptive machine. We will sometimes use the notation $M^A(x) = 0$ for rejecting, and $M^A(x) = 1$ for accepting computations.

For a Turing machine M, $L(M)$ denotes the set of strings accepted by M. For an oracle machine M and set A, $L(M, A)$ denotes the set of strings accepted by M relative to oracle A. These sets are also called the *language* of M and the language of M^A respectively.

To obtain model independent results we consider only space classes which are closed under constant factor space overhead, and time classes which are closed under polynomial time overhead.

2.2 Truth tables

The pair $<<a_1, \ldots, a_k>, \alpha>$ is called a *truth-table condition of norm* k if $<a_1, \ldots, a_k>$ is a k-tuple ($k > 0$) of strings, and α is a k-ary Boolean function [11]. The set $\{a_1, \ldots, a_k\}$ is called the *associated set* of the tt-condition. A function f is a *truth-table function* if f is total and $f(x)$ is a truth-table condition for every x in Σ^*. If, for all x, $f(x)$ has norm less than or equal to k, then f is called a k-truth-table ($k - tt$) function. If a function f is a

k-tt function for some integer ($k > 0$) then we call f a ***bounded truth-table*** (btt) function. We say that a tt-function f is a ***disjunctive*** truth-table (dtt) function if f is a truth-table condition whose Boolean function is always disjunctive.

2.3 Reductions, reducibilities and completeness

Let the resource bound b be either polynomial-time or logarithmic space and $A_1, A_2 \subseteq \Sigma^*$. We say that:

1. A_1 is b many-on reducible to A_2 ($\leq_m^b$-reducible) iff there exists a function f computable within resource bound b such that $x \in A_1$ iff $f(x) \in A_2$.

2. A_1 is b truth-table reducible ($\leq_{tt}^b$-reducible) to A_2 iff there exists a b-bounded tt-function f such that $\alpha(\chi_{A_2}(a_1), \ldots, \chi_{A_2}(a_k)) = \textbf{true}$ iff $x \in A_1$, where $f(x)$ is $<<a_1, \ldots, a_k>, \alpha>$ and χ_{A_2} is the characteristic function of the set A_2. As b-bounded functions can be computed by b-bounded Turing machines, the truth table conditions are often modeled by non-adaptive oracle machines. Resource bounded k-truth-table reductions ($\leq_{k-tt}^b$) and bounded-truth-table reductions ($\leq_{btt}^b$) are defined similarly using $k - tt$ and btt functions.

3. A_1 is b Turing reducible to A_2 ($\leq_T^b$-reducible) to A_2 if there exists a b-bounded deterministic oracle machine such that $A_1 = L(M, A_2)$.

4. A_1 is b disjunctive reducible ($\leq_d^b$-reducible) to A_2, if $A_1 \leq_d^b A_2$ by some dtt-function. For $k > 0$, A_1 is k-disjunctive reducible ($\leq_{k-d}^b$) to A_2, if $A_1 \leq_{tt}^b A_2$ by some dtt-function of norm k.

Let $\leq_r^b$ be any of the above reductions

1. A set A is $\leq_r^b$ hard for some complexity class C iff for all $B \in C$, B is $\leq_r^b$ reducible to A.

2. A set A is $\leq_r^b$ complete for some complexity class C iff A is $\leq_r^b$ hard for C and $A \in C$.

   For any complexity class C, a set A is C-immune iff no infinite $L \in C$ is a subset of A.

3 Nondeterministic Time Classes

We first turn our attention to nondeterministic time classes. Our results will apply to any nondeterministic time class which contains $NE = \cup_{c \in I\!N} \text{NTIME}(2^{cn})$. Similar results were proved for deterministic classes by Watanabe [13]. The nondeterministic classes are not known to be closed under complement, and so a quite different approach is needed.

The first result exhibits a difference between complete sets with respect to $\leq_m^p$ and $\leq_d^p$ (and also $\leq_m^{logspace}$ and $\leq_d^{logspace}$) reductions. The new tool that is needed comes from the work of Ganesan and Homer [5].

First a definition.

Definition 3.1 *A function g is said to be exponentially honest if for all x : $2^{|g(x)|} \geq |x|$*

The next result appears in theorem 3 in [5]

Theorem 1 *Any $\leq_m^p$ complete set for NE is one-one complete via functions which are exponentially honest. That is, if C is $\leq_m^p$-complete for NE then for any $B \in NE$ there is an exponentially honest, 1-1, polynomially-time computable function f which many-one reduces B to C.*

While the major new contribution of this theorem was the one-one completeness, it is the exponential honesty which will be crucial here. In order to get our results concerning logspace reductions we state a slight strengthening of this theorem. First note that $K = \{<e, x, l>|\text{the } e^{th} NE \text{ machine accepts } x \text{ in } \leq l \text{ steps }\}$ is $\leq_m^{logspace}$ complete for NE. Given this, the following theorem has essentially the same proof as the previous theorem and so its proof will be omitted here.

Theorem 2 *Any $\leq_m^{logspace}$-complete set for NE is one-one complete via functions which are exponentially honest.*

We can now state our theorem which yields the desired differences between complete sets. A similar theorem, slightly weaker as it applies only to polynomial-time reductions, can be found in Ganesan and Homer [5]. The proof presented here is simpler, more complete and will be generalized to other reducibilities later in this section.

Theorem 3 *There is a set B which is $\leq_{2-d}^{logspace}$-complete for* NTIME $\left(2^{poly}\right)$ *but not $\leq_m^p$-complete for* NTIME $\left(2^{poly}\right)$.

Proof: Let K be the $\leq_m^{logspace}$-complete set for NE defined above. It is easy to see (cf. Balcázar, Díaz, and Gabarró [1]) that K is $\leq_m^{logspace}$-complete for NTIME $\left(2^{poly}\right)$ as well. The set B will be constructed so that its only elements are of the form $<e, x, l, i>$, $i = 0$ or $i = 1$.

B will be complete via the $\leq_{2-d}^{logspace}$ reduction:

$$<e, x, l> \in K \leftrightarrow [<e, x, l, 0> \in B] \vee [<e, x, l, 1> \in B]$$

To ensure that B is not $\leq_m^p$-complete we diagonalize against all possible $\leq_m^p$ reductions from $\Sigma^*$to B. Let f_i be the i^{th} polynomial-time computable function in some fixed enumeration of all such functions. We may assume that f_i runs in DTIME $\left(n^i\right)$. We need a set of elements on which to diagonalize. To this end we define a sequence of integers $\{u_n\}_n$ by $u_0 = u_1 = 1$, $u_m = 2^{(u_{m-1})^{m-1}} + 1$, for $m > 1$.

Let $H = \{0^{u_k}\}_{k \in \mathbb{N}}$. It is easy to verify that $H \in P$. We use the sequence H to diagonalize against $\leq_m^p$ reductions

We can now describe the construction of B. The set B is constructed in stages. At stage $k = 1, 2, ...$ we determine all elements in B of length $\leq (u_k)^k$. At stage 1 we put all strings s, $|s| \leq 1$ into $\overline{B}$. Now assume we have constructed B through stage $n - 1$ and describe stage $n > 1$.

stage n:

Compute $f_n(0^{u_n})$. Let s be any string of the form $<e, x, l, i>$, $(i \in \{0, 1\})$ with $(u_{n-1})^{n-1} < |s| \leq (u_n)^n$. Then we put $s \in B$ iff $s \neq f_n(0^{u_n})$ and $<e, x, l> \in K$.

end of stage n

First note that $K \leq_{2-d}^{logspace} B$ via the reduction defined above. Since for any $<e,x,l>$ if $<e,x,l> \in K$ then at least one of $<e,x,l,0>, <e,x,l,1>$ is put into B (without loss of generality $|<e,x,l,0>| = |<e,x,l,1>|$) and if $<e,x,l> \notin K$ then neither of the two strings is in B.

CLAIM 3.1 $B \in \mathrm{NTIME}\left(2^{poly}\right)$

Proof: Given a string s, $s \in B$ iff:

1. $s = <e,x,l,i>$ for some $e,x,l \in \Sigma^*, i \in \{0,1\}$,
2. $<e,x,l> \in K$, and
3. $s \neq f_k(0^{u_k})$ where u_k is the least element in the sequence $\{u_n\}_n$ with $(u_k)^k \geq s$.

1. can be tested for in linear time. Consider 3. By definition of u_k, $|s| > (u_{k-1})^{k-1}$ and hence $(u_k)^k \leq 2^{k|s|}$. Now since $f_k \in \mathrm{DTIME}\left(n^k\right)$ and $H \in P$, the u_k as in 3 can be found and the condition in 3 checked in $(u_k)^k \leq 2^{k|s|} \leq 2^{O\left(|s|^2\right)}$ steps. As $K \in \mathrm{NTIME}\left(2^n\right)$ the claim follows and in fact $B \in \mathrm{NTIME}\left(2^{n^2}\right)$. ⊠

Thus we have $B \leq_{2-d}^{logspace}$-complete for $\mathrm{NTIME}\left(2^{poly}\right)$

CLAIM 3.2 *B is not $\leq_m^p$-complete for* $\mathrm{NTIME}\left(2^{poly}\right)$.

Proof: Assume B were $\leq_m^p$-complete. Then by theorem 2 there is a polynomial time computable f_n which reduces $\Sigma^*$to B and which is exponentially honest.

At stage n of the construction of B we computed $f_n(0^{u_n})$. By the exponential honesty of f_n, $2^{|f_n(0^{u_n})|} \geq |0^{u_n}| = u_n = 2^{(u_{n-1})^{n-1}} + 1$, and so $|f_n(0^{u_n})| > (u_{n-1})^{n-1}$. Hence at stage n we put $f_n(0^{u_n})$ into $\overline{B}$. This contradicts the assumption that f_n is a reduction of $\Sigma^*$to B ⊠

This completes the proof of theorem 3 ⊠

A standard padding argument now yields the same result for NE.

Corollary 1 *There is a C which is $\leq_{2-d}^{logspace}$-complete for NE but not $\leq_m^p$-complete for NE.*

Proof: Let B be as in the previous theorem. Then, as noted above, $B \in \mathrm{NTIME}\left(2^{n^2}\right)$. Define $C = \{x10^{|x|^2} | x \in B\}$. Then

1. $C \in NE$
2. $B \leq_m^{logspace} C$ and hence C is $\leq_{2-d}^{logspace}$ complete.
3. C is not $\leq_m^p$ complete for NE. $(C \leq_m^{logspace} B)$

Hence C has the desired properties. ⊠

It follows directly from the corollary that there are $\leq_{2-d}^{logspace}$-complete sets ($\leq_{2-d}^{p}$-complete sets) for NE which are not $\leq_{m}^{logspace}$-complete ($\leq_{m}^{p}$-complete) for NE. Furthermore the same proof works to give these same results for any nondeterministic class containing NE, including the class of recursively enumerable sets.

We next turn to differentiating between complete sets for bounded truth table reductions. We will prove that, for any $k > 1$ there is a set which is $\leq_{k-tt}^{p}$-complete, but not $\leq_{(k-1)-tt}^{p}$-complete for NE.

A similar result holds for $\leq_{k-tt}^{logspace}$-complete sets as well. For simplicity, we present the proof for the case $k = 3$.

The general theorem is a direct extension of the proof given here. The central idea in the proof is again a careful analysis of the honesty of the reductions. However, here we cannot avoid reductions which are not exponentially honest. Rather, we show that in exponential time, we can directly compute the result of dishonest queries made by the reduction as they are so much shorter than the input. Honest queries made by the reduction are handled as in theorem 3.

Theorem 4 *There is a set B which is $\leq_{3-d}^{p}$-complete for* NTIME $\left(2^{poly}\right)$ *but not $\leq_{2-tt}^{p}$-complete for* NTIME $\left(2^{poly}\right)$

Note that this theorem separates both $\leq_{2-tt}^{p}$ and $\leq_{3-tt}^{p}$-completeness and $\leq_{2-d}^{p}$ completeness from $\leq_{3-d}^{p}$-completeness.

Proof: The set B is constructed in stages, in a way similar to that of theorem 3. B will be made $\leq_{3-d}^{p}$-complete via the reduction.

$$<e,x,l> \in K \leftrightarrow \exists i \in \{0,1,2\} : <e,x,l,i> \in B$$

In order to ensure that B is not $\leq_{2-tt}^{p}$-complete we simultaneously construct a set W which witnesses the incompleteness of B. We make use of the sequence $\{u_n\}$ from the previous proof. Recall that $u_k = 2^{(u_{k-1})^{k-1}} + 1$.

Let M_i be the i^{th} 2-tt reduction in some enumeration of such reductions and let $Q(M_i, x)$ be the set of (at most two) elements queried by $M_i^S(x)$ during its computation. M_i is assumed to run in time $p_i(n) = n^i$ and, as M_i is a truth-table reduction $Q(M_i, x)$ does not depend on S. We can now present the construction of B and W.

Initially $B = W = \emptyset$.

stage n:

At stage n we determine $B(y)$ for all strings y with $(u_{n-1})^{n-1} < |y| \leq (u_n)^n$ and we decide whether or not $0^{u_n} \in W$. (At this point in the construction $B \subseteq \Sigma^{\leq (u_{n-1})^{n-1}}$.)

1. For all y with $(u_{n-1})^{n-1} < |y| \leq (u_n)^n$, put

$$y \in B \leftrightarrow \begin{cases} y \notin Q(M_n, 0^{u_n}) & \text{and} \\ \exists i \in \{0,1,2\}(y = <e,x,l,i>) & \text{and} \\ <e,x,l> \in K & \end{cases}$$

2. Put $0^{u_n} \in W \leftrightarrow M_n^B(0^{u_n}) = 0$

end of stage n

Clearly, as $|Q(M_n, 0^{u_n})| \leq 2$, $<e,x,l> \in K \leftrightarrow \exists i \in \{0,1,2\}(<e,x,l,i> \in B$. So B is $\leq^p_{3-d}$-hard for NTIME $\left(2^{poly}\right)$. Moreover since only elements not in $Q(M_n, 0^{u_n})$ are added to B at stage n, and only elements of length greater than $(u_n)^n$ at subsequent stages $M_n^B(0^{u_n}) = M_n^{B^{\leq (u_{n-1})^{n-1}}}(0^{u_n})$

We proceed via a series of lemmata to complete the proof.

Lemma 4.1 $W \in$ DTIME $\left(2^{poly}\right)$

Proof: By the construction $W \subseteq \{0^{u_n}\}$. The following algorithm tests if $0^{u_n} \in W$

1. compute $Q(M_n, 0^{u_n})$.

2. For each $y \in Q(M_n, 0^{u_n})$, compute if $y \in B$ as follows:
   if $|y| > (u_{n-1})^{n-1}$ then $y \notin B$ by the construction.
   if $|y| \leq (u_{n-1})^{n-1}$ then find the least k such that $|y| \leq (u_k)^k$;
   if $y \in Q(M_k, 0^{u_k})$ then $y \notin B$
   else $y \in B \leftrightarrow y = <e,x,l,i>$ for some $i \in \{0,1,2\}$ and $<e,x,l> \in K$

3. Using the truth table computed by $M_n^B(0^{u_n})$ and the information from 2, we can compute the value of $M_n^B(0^{u_n})$. By the construction $0^{u_n} \in W$ if and only if $M_n^B(0^{u_n}) = 0$.

Now 1. takes at most $(u_n)^n$ steps. For step 2, given 0^{u_n} as input, we can in u_n steps determine $(u_{n-1})^{n-1}$ since $u_n = 2^{(u_{n-1})^{n-1}}+1$. If $|y| \leq (u_{n-1})^{n-1}$, then finding k least with $(u_k)^k \geq |y|$ can again be done within u_n steps. (by the definition of the sequence $\{u_n\}$.). We can then compute $Q(M_k, 0^{u_k})$ in $(u_k)^k < (u_n)^n$ steps and test if $y \in Q(M_k, 0^{u_k})$. If so and if $y = <e,x,l,i>$ then computing if $<e,x,l> \in K$ deterministically takes $2^{2^{|<e,x,l>|}} \leq 2^{2^{(u_{n-1})^{n-1}}} < 2^{u_n}$ steps. Hence step 2 can be carried out in time $2^{u_n} + (u_n)^n \in 2^{O(u_n)}$.

Finally step 3 can be done in $O\left((u_n)^n\right)$ steps. So deciding $0^{u_n} \in W$ takes $2^{O(u_n)})$ steps and hence $W \in$ DTIME $\left(2^{poly}\right)$. ⊠

Lemma 4.2 *B is $\leq^p_{3-d}$-complete for* NTIME $\left(2^{poly}\right)$.

Proof: We have already observed that B is $\leq^p_{3-d}$-hard. So it remains to prove that $B \in$ NTIME $\left(2^{poly}\right)$.

Given y, the following algorithm tests if $y \in B$.

1. Find the least n with $|y| \leq (u_n)^n$.

2. Compute $Q(M_n, 0^{u_n})$.

3. If $y \in Q(M_n, 0^{u_n})$ then $y \notin B$
   else

$$y \in B \leftrightarrow \begin{cases} y = <e,x,l,i> & \text{for some } i \in \{0,1,2\} \\ \text{and } <e,x,l> \in K & \end{cases}$$

Now, for n as in 1, $|y| > (u_{n-1})^{n-1}$, so by definition of the $\{u_n\}$ sequence $(u_n)^n = \left(2^{(u_{n-1})^{n-1}} + 1\right)^n < \left(2^{|y|} + 1\right)^n < 2^{n|y|+1}$. Hence, since $n < |y|$, $(u_n)^n < 2^{O(|y|^2)}$, the value of n in 1 can be found in $< (u_n)^n$ steps and 2 can be computed in $(u_n)^n$ steps. Clearly then 3 nondeterministically computed in time $2^{|y|}$ as $|<e,x,l>| < |y|$.

Thus steps 1 and 2 can be carried out deterministically in $2^{O(|y|^2)}$ steps and step 3 can be done in NTIME $\left(2^{poly}\right)$, so $B \in$ NTIME $\left(2^{poly}\right)$. ⊠

Lemma 4.3 *B is not $\leq^p_{2-tt}$-complete for* NTIME $\left(2^{poly}\right)$.

Proof: Assume B were $\leq^p_{2-tt}$-complete. Then by lemma 4.1, W would be 2-tt reducible to B, say by reduction M_n. But by the construction, we have $0^{u_n} \in W$ if and only if $M_n^B(0^{u_n}) = 0$, contradicting the assumption that M_n is the required reduction. ⊠

⊠

Via the same padding argument as before one can prove.

Corollary 2 *There is a set C which is $\leq^p_{3-d}$-complete for NE, but not $\leq^p_{2-tt}$-complete for NE.*

Straightforward modifications of the above method yield a number of extensions of these results.

- The results are true for logspace reductions rather than polynomial time reductions.
- The above proof can be generalized to give the same results for k-d reductions instead of 3-d.
- The above proofs work as well for any nondeterministic class with a paddable $\leq_m$-complete set which contains *NE*.

A very similar argument can be used to separate $\leq^p_{btt}$ and $\leq^p_{tt}$ completeness. The proof is only sketched.

Theorem 5 *There is a set B which is $\leq^p_{tt}$-complete for* NTIME $\left(2^{poly}\right)$, *but not $\leq^p_{btt}$-complete for* NTIME $\left(2^{poly}\right)$.

Proof:(Sketch): As before we construct B together with a witness set W. The reduction making B $\leq^p_{tt}$-complete will be:

$$<e,x,l> \in K \leftrightarrow \exists i(i \leq |<e,x,l>| \text{ and } <e,x,l,i> \in K)$$

At stage n of the construction we treat n as a pair $n = <n_1, n_2>$ and we try to diagonalize against the n_1^{st} tt-reduction M_{n_1}, but we only do this if M_{n_1} asks $\leq n_2$ queries to the oracle.

More formally, stage n of the construction is as follows:

stage n:

1. Let $n = <n_1, n_2>$ and compute $Q(M_{n_1}, 0^{u_n})$.

2. If $|Q(M_{n_1}, 0^{u_n})| > n_2$ then
   for for all y with $(u_{n_1})^{n-1} < |y| \leq (u_n)^n$, put
   $y \in B \leftrightarrow y = <e, x, l, 0>$ and $<e, x, l,> \in K$.

3. If $|Q(M_n, 0^{u_n})| \leq n_2$ then

4. put $0^{u_n} \in W \leftrightarrow M_{n_1}^B(0^{u_n}) = 0$ and

5. for all y with $(u_{n-1})^{n-1} < |y| \leq (u_n)^n$ put

$$y \in B \leftrightarrow \begin{cases} y \notin Q(M_n, 0^{u_n}) & \text{and} \\ \exists i(y = <e, x, l, i> \text{ and } i \leq |<e, x, l>| & \text{and} \\ <e, x, l>) \in K & \end{cases}$$

end of stage n

Now exactly as in theorem 4, we can prove that $W \in \text{DTIME}\left(2^{poly}\right)$ and that $B \in \text{NTIME}\left(2^{poly}\right)$. In step 5 of the construction we have that $(u_{n-1})^{n-1} < |y|$ so it follows from the definition of $\{u_n\}$ that if $y = <e, x, l>$ and $i \leq |<e, x, l>|$ then $|<e, x, l>| > n > n_2$. So in step 5 we have room to code K into B. Hence B is $\leq_{tt}^p$ complete for $\text{NTIME}\left(2^{poly}\right)$.

Now, let M_{n_1} be a $\leq_{btt}^p$-reduction, say with norm n_2. Then at stage $n = <n_1, n_2>$ we will find that $|Q(M_{n_1}, 0^{u_n})| \leq n_2$ and so in step 3 of the construction will put $0^{u_n} \in W \leftrightarrow M_{n_1}^B(0^{u_n}) = 0$, and so 0^{u_n} will witness the fact that M_{n_1} does not btt-reduce W to B and so B is not $\leq_{btt}^p$-complete for $\text{NTIME}\left(2^{poly}\right)$. ⊠

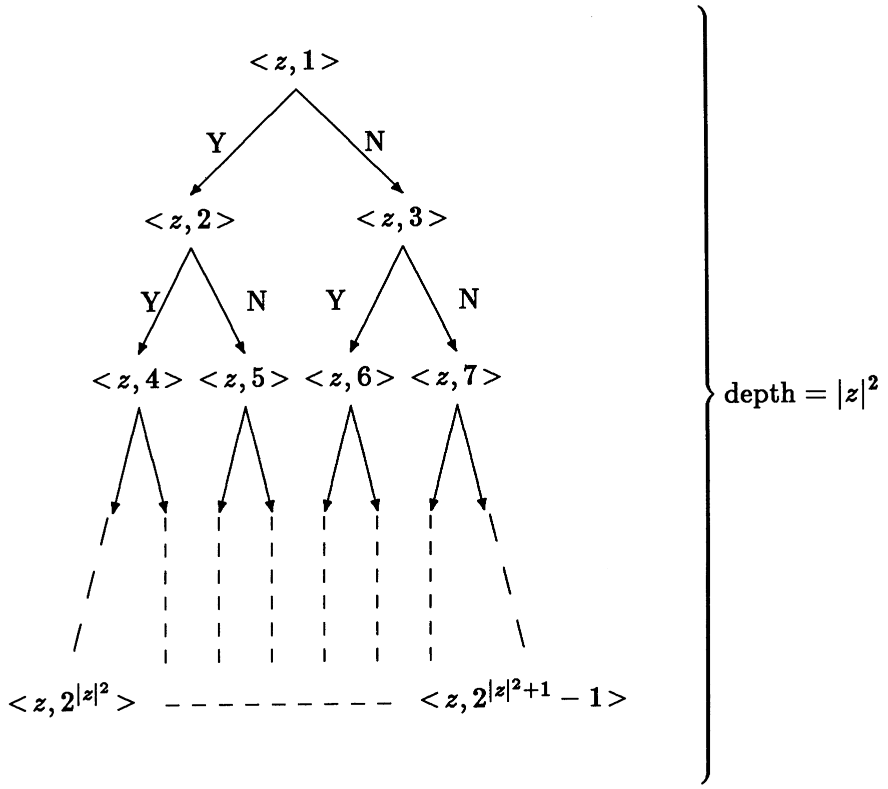
Finally, we want to separate tt-completeness from Turing completeness on NE. While the underlying ideas are the same, the construction is considerably more complex. The key is to define the correct Turing reduction within which there is room to diagonalize against all tt-reductions.

Theorem 6 *There is a set B which is $\leq_T^p$-complete but not $\leq_{tt}^p$-complete for* $\text{NTIME}\left(2^{poly}\right)$.

Proof: First we define the Turing reduction which we will use to show B $\leq_T^p$-complete.

On any input z, the Turing reduction relative to B asks a series of $|z|^2$ queries. Each query is of the form $<z, i> \in B$?, where $i < 2^{|z|^2}$. For convenience we require that all queries have the same length. We do this by padding each i in the pair $<z, i>$ with enough leading 0's so that $|<z, i>| = |<z, 2^{|z|^2+1} - 1>|$. So the length of each query $<z, i>$ is $|z| + |z|^2$. Which queries are asked depends on the answers to the previous query. The first string queried is $<z, 1>$. If $<z, 1> \in B$ then $<z, 2>$ is queried next, if $<z, 1> \notin B$ then $<z, 3>$ is the next query. More generally, when $<z, i>$ is queried of B a "yes" answer results in $<z, 2i>$ being the next query and a "no" answer in $<z, 2i + 1>$ being the next query. This process continues on for $|z|^2$ many queries. The reduction then halts and accepts z iff the number of "yes" answers in the $|z|^2$ many queries was odd.

A picture of the query tree of this reduction on input z is:



Note that the number of nodes in this tree is $2^{|z|^2+1} - 1$ and the number of leaves is $2^{|z|^2}$. Let $T^B(z)$ be this Turing reduction.

The property we will preserve, and which will guarantee $\leq_T^p$-completeness of B is that $z \in K$ if and only if $T^B(z)$ accepts.

Some terminology will be helpful here in order to simplify the construction. For any z, let R_z be the query tree for the reduction $T^B(z)$ described and pictured above, and let N_z be the collection of nodes in R_z. Assume we are given a set $S \subseteq N_z$ with $|S| < 2^{|z|^2-1}$. Then as there are $2^{|z|^2-1}$ pairs of leaves in R_z, each of the form $<z, 2i>, <z, 2i+1>$, there is some such pair neither of which are in S. Let $\ell(S)$ be the lexicographically least leaf in $N_z - S$ such that $p(S)$, the path from $<z,1>$ to $\ell(S)$ in R_z, has the property that the number of "yes" edges on $p(S)$ is even. (Note: such an $\ell(S)$ and $p(S)$ must exist since both $<z, 2i>$ and $<z, 2i+1>$ are in $\overline{S}$ and one of the paths $<z,1> \ldots <z,2i>$ or $<z,1> \ldots <z, 2i+1>$ must contain an even numer of "yes" edges.)

The construction will work as follows. For each z we find a particular set $S \subseteq N_z$ as above and corresponding path $p(S)$. We then define B on elements $<z, i>$ so that the reduction $T^B(z)$ follows the path $p(S)$ through R_z. Finally we put $\ell(S) \in B \leftrightarrow z \in K$. This wil ensure that

$$\begin{aligned} z \in K &\leftrightarrow \text{The number of "yes" edges on } p(S) \text{ is odd} \\ &\leftrightarrow T^B(z) \text{ accepts} \end{aligned}$$

As in the previous proofs we simultaneously construct B and a witness set W. We again use the sequence $\{u_k\}$. However, to diagonalize against the n^{th} tt-reduction M_n, we find a suitable point $0^{v_n} \in \{0^{u_k}\}$ and diagonalize at 0^{v_n}.

It is simplest to define the sequence $\{v_n\}$, and prove it exists before presenting the actual construction. Let $v_0 = 0$ and assume v_{n-1} has been defined. Then v_n is the smallest element u_t in the sequence $\{u_k\}$ such that

1. $u_t > v_{n-1}$ and
2. $(u_t)^t < 2^{(u_{t-1})^{2(t-1)}-1}$

CLAIM 6.1 *A sequence $\{v_n\}$ exists as defined above.*

Proof: By definition of $\{u_k\}$, $u_k = 2^{(u_{k-1})^{k-1}} + 1$, so

$$\begin{aligned}(u_k)^k &= \left(2^{(u_{k-1})^{k-1}} + 1\right)^k \\ &< \left(2^{(u_{k-1})^{k-1}+1}\right)^k \\ &= 2^{k(u_{k-1})^{k-1}+k}\end{aligned}$$

Since the sequence $\{u_k\}$ increases exponentially, $k << u_{k-1}$, and so for k sufficiently large $2^{k(u_{k-1})^{k-1}+k} < 2^{(u_{k-1})^{2(k-1)}-1}$ as needed. ⊠

Initially $B = W = \emptyset$. We can now define stage n of the construction. At stage n we decide whether to put 0^{v_n} into W and we define B on all y with $(v_{n-1})^{n-1} < |y| \leq (v_n)^n$.
stage n:
At this point $B \subseteq \Sigma^{\leq (v_{n-1})^{n-1}}$.

1. Put $0^{v_n} \in W \leftrightarrow M_n^B(0^{v_n}) = 0$.
2. By definition of v_n, $v_n \in \{u_k\}$, say $v_n = u_t$. For all y such that $(v_{n-1})^{n-1} < |y| \leq (u_{t-1})^{t-1}$, put
$$y \in B \leftrightarrow \begin{cases} y = \, <e, x, l, 2^{|<e,x,l>|^2+1} - 1> \\ \text{and } <e, x, l> \in K \end{cases}$$
3. For all y such that $(u_{t-1})^{t-1} < |y| \leq (v_n)^n$, if $y = \, <e, x, l, i>$ with $i < 2^{|<e,x,l>|^2+1}$ then set $S = Q(M_n, 0^{v_n}) \cap N_{<e,x,l>}$. Compute $\ell(S)$ and $p(S)$ as defined above. (Note: we will prove later that $p(S)$ and $\ell(S)$ exist.) Then put $y \in B$ only if either
   (a) $y \in p(S)$ and $y \neq \ell(S)$ and $<e, x, l, 2i> \in p(S)$ or
   (b) $y \in p(S)$ and $y = \ell(S)$ and $<e, x, l> \in K$

end of stage n

We first prove that $K \leq_T^p B$ via the reduction procedure T^B defined above. Given $<e, x, l>$, let $y = \, <e, x, l, 2^{|<e,x,l>|^2+1} - 1>$ and let n be such that $(v_{n-1})^{n-1} < |y| \leq (u_n)^n$. There are 2 cases. If $(v_{n-1})^{n-1} < |y| \leq (u_{t-1})^{t-1}$ then part 2 applies to y, and to all strings in the tree $R_{<e,x,l>}$. In this case, the only possible element of the computation query tree of $T^B(<e, x, l>)$ which is put into B is y itself. So the computation $T^B(<e, x, l>)$ receives "no" answers to all of its queries until it queries y. By part 2 we have $<e, x, l> \in K \leftrightarrow y \in B \leftrightarrow$ the computation of $T^B(<e, x, l>)$ receives an odd number (exactly 1) of "yes" answers $\leftrightarrow T^B(<e, x, l>)$ accepts.

In the second case $(u_{t-1})^{t-1} < |y| \le (v_n)^n$ and part 3 of the construction applies to y, and to all queries in the tree $R_{<e,x,l>}$. Now note that, since M_n runs in time $p_n(t) = t^n$, $|S| = |Q(M_n, 0^{v_n})| \le (v_n)^n$. By choice of v_n, $(v_n)^n < 2^{(u_{t-1})^{2(t-1)}-1} \le 2^{|y|^2}$. Putting this toghether gives $|S| \le 2^{|y|^2-1}$, so by the discussion at the beginning of the proof, the quantities $\ell(S)$ and $p(S)$ in part 3 are well defined. Then in part 3 we put elements of $p(S)$ into B or $\overline{B}$ in such a way that $T^B(<e,x,l>)$ follows the path $p(S)$ to the leaf $\ell(S)$. And by part 3b we have $<e,x,l> \in K \leftrightarrow \ell(S) \in B \leftrightarrow T^B(<e,x,l>)$ has an odd number of "yes" queries $\leftrightarrow T^B(<e,x,l>)$ accepts.

Hence in either case the reduction works and we have $K \le_T^p B$.

We now proceed to finish the proof via a series of three lemmata.

Lemma 6.1 $W \in \text{DTIME}\left(2^{poly}\right)$

Proof: Since $\{O^{v_n}\}$ can easily be seen to be in P, given a string x we can check if $x = O^{v_n}$ for some n. Then given 0^{v_n} we can determine if $0^{v_n} \in W$ by computing $M_n^B(0^{v_n})$ as follows.

1. Compute $Q(M_n, 0^{v_n})$
2. For each $y \in Q(M_n, 0^{v_n})$, we determine if $y \in B$ using 3 and 4 below.
3. If $|y| \le (u_{t-1})^{t-1}$ and for some e, x, l, i with $i < 2^{|<e,x,l>|^2+1}$, $y = <e,x,l,i>$, then we compute if $y \in B$ directly using parts 2 and 3 of the construction. To do this we compute the relevant $v_{n'}$ and $u_{t'-1}$ which bound $|y|$ and, if needed, the corresponding $Q(M_{n'}, 0^{v_{n'}}), S, p(S), \ell(S)$ and deterministically compute if $<e,x,l> \in K$ if part 3b of the construction applies.
4. If $(u_{t-1})^{t-1} < |y| \le (v_n)^n$ and for some e, x, l, i with $i < 2^{|<e,x,l>|^2+1}$, $y = <e,x,l,i>$ then we use part 3 to decide if $y \in B$. Compute $S = Q(M_n, 0^{v_n}) \cap N_{<e,x,l>}$, $\ell(S)$ and $p(S)$. Note that in this case $y \ne \ell(S)$ since $\ell(S) \notin Q(M_n, 0^{v_n})$. So 3a applies and we have $y \in B \leftrightarrow y \in p(S)$ and $<e,x,l,2i> \in p(S)$.

It is straightforward to check that all of the above can be computed in DTIME $\left(2^{poly}\right)$ relative to $v_n = |0^{v_n}|$. The key point is that when, in 3 above, $<e,x,l> \in K$ is computed, we have $|<e,x,l>| < |y| \le (u_{t-1})^{t-1}$. Now $<e,x,l> \in K$ can be determined in time $2^{2^{|<e,x,l>|}} \le 2^{2^{(u_{t-1})^{t-1}}} < 2^{v_n}$. Finally, once all of $Q(M_n, 0^{v_n})$ is determined, we have $0^{v_n} \in W \leftrightarrow M_n^B(0^{v_n}) = 0$. ⊠

Lemma 6.2 $B \in \text{NTIME}\left(2^{poly}\right)$

Proof: The proof is very similar to that of Lemma 6.1. Given y, we want to compute if $y \in B$. We do this just as we determined in Lemma 6.1. The added complexity comes from the fact that here part 3b of the proof may apply. It may be that $y = \ell(S) \in p(S)$ and so $y \in B \leftrightarrow <e,x,l> \in K$ in this case. This puts B into NTIME $\left(2^{poly}\right)$. The other cases for deciding B can all be carried out in DTIME $\left(2^{poly}\right)$ as in Lemma 6.1 ⊠

Lemma 6.3 *B is not $\leq_{tt}^{p}$-complete for* NTIME $\left(2^{poly}\right)$

Proof: If B were $\leq_{tt}^{p}$-complete, we would have $W \leq_{tt}^{p} B$, say via reduction procedure M_n. But 0^{v_n} witnesses that this is not the case. ⊠

⊠

Corollary 3 *There is a set C which is $\leq_{T}^{p}$-complete for NE, but not $\leq_{tt}^{p}$-complete for NE*

Proof: Again a simple padding argument works. Let B be as in Theorem 6. Define $C = \{x10^{|x|^2} | x \in B\}$. Since $B \in$ NTIME $\left(2^{n^2}\right)$, $C \in NE$, $B \leq_{m}^{p} C$, and hence C is $\leq_{T}^{p}$-complete for NE. Now assume C were $\leq_{tt}^{p}$-complete for NE. Then $W \leq_{tt}^{p} C$, where W is the set defined in the proof of Theorem 6. Let M be a $\leq_{tt}^{p}$-reduction which reduces W to C. Define a $\leq_{tt}^{p}$ reduction M_0 as follows. If $M^{C}(x)$ generates the tt-condition $<<y_1, y_2, \ldots, y_k>, \alpha>$, then $M_0^{B}(x)$ generates the condition $<<z_1, z_2, \ldots, z_k>, \alpha>$, where $z_i =$ (if $y_i = z10^{|z|^2}$ then z else b) and b is a fixed element of $\overline{B}$. Clearly M_0 is a $\leq_{tt}^{p}$-reduction and M_0 reduces W to B, contradicting the proof of theorem 6. ⊠

It is not clear here how to modify the above proof to make $B \leq_{T}^{logspace}$-complete for NTIME $\left(2^{poly}\right)$. In particular it is not clear how to use only logtape to prove the $\leq_T$-completeness of B as was done in the above proof. Watanabe [14] used Kolmogorov complexity to differentiate between $\leq_{tt}^{p}$ and $\leq_{T}^{p}$ completeness for $DEXT$. The proof of theorem 6 would work for $DEXT$ as well and does not appeal to Kolmogorov complexity.

4 Nondeterministic Space Classes

We now turn our attention to nondeterministic space classes. Our results will apply to any such class properly containing $NLOGSPACE = \cup_{c \in \mathbb{N}} NSPACE(c \log n)$. The reason for demanding proper containment of $NLOGSPACE$ is that in the following constructions we want to diagonalize against $\leq_{r}^{logspace}$ reductions. Yet we want

the language resulting from the diagonalization to be complete for some nondeterministic space class. Therefore the results in the sequel will be about classes $NSPACE(S(n))$ where it is understood that $S(n)$ is some fully space constructible function with the property that: $\lim n \rightarrow \infty \log(n)/S(n) = 0$

As in the case of nondeterministic time classes the property needed for differentiating between reductions is their honesty. Using a lemma with a similar statement as that in Watanabe [13] and with the observation that nondeterministic space classes above $LOGSPACE$ are closed under complementation, we get an especially nice form of honesty for logspace reducibilities on these classes.

As it turns out all logspace reducibilities on these classes have some length increasing property. As "length increasing" is not an unambiguous term for say $\leq_{k-tt}^{logspace}$-reductions, we will first define precisely what this length increasing property is. A *function f* will be called *length increasing* iff $\overset{\infty}{\forall} x : |f(x)| > |x|$. Now:

Definition 4.1 *Let M be a deterministic logarithmic-space bounded oracle machine and let A be an oracle set such that M witnesses a $\leq_{r}^{logspace}$-reduction.*

1. *We say that a function f is* generated by M *and* A *iff* f *maps almost all* $x \in \Sigma^*$ *to some element of* $Q(M, A, x)$.
2. Fm = $\{f \mid f$ *is generated by a logarithmic-space bounded oracle machine which corresponds to some* $\leq_m^{logspace}$*-reduction*$\}$.
3. $Fbtt$ = $\{f \mid f$ *is generated by a logarithmic-space bounded oracle machine which corresponds to some* $\leq_{btt}^{logspace}$*-reduction*$\}$.
4. Ftt =$\{f \mid f$ *is generated by a logarithmic-space bounded oracle machine which corresponds to some* $\leq_{tt}^{logspace}$*-reduction*$\}$
5. FT = $\{f \mid f$ *is generated by a logarithmic-space bounded oracle machine which corresponds to some* $\leq_T^{logspace}$*-reduction*$\}$
6. *Let* Fr *be any of the above classes. A set* A *has an* Fr*-subset if there exists a function* $f \in Fr$, *which is total and length increasing, such that for almost all* $x \in \Sigma^*$, $f(x) \in A$.

We will show in the next subsection that languages complete in $NSPACE(S(n))$ under $\leq_r^{logspace}$ reductions have a *Fr*-subset, and are in that sense complete under an honest reduction.

4.1 Structure of complete sets in NSPACE

The first theorem we derive on logspace reductions on $NSPACE(S(n))$ states that for a complete set in $NSPACE(S(n))$ there can always be found a set in $NSPACE(S(n))$ which reduces to the complete set via a length increasing reduction (with l.i. defined as in Definition 4.1). A similar theorem (but for $DEXT$) can be found in Watanabe [13]. It is not obvious that a uniform estimate (beneath something like polylogarithmic time) on the space usage of logarithmic space bounded machines can be given as could be done in section 3. Therefore we have to treat the case where the simulation threatens to use more that $S(n)$ space separately.

Theorem 7 *Let* A *be any set in* $NSPACE(S(n))$.
There exists a set $L_A \in NSPACE(S(n)) \subseteq N \times \Sigma^*$, *such that if* $L_A \leq_T^{logspace} A$ *via some polynomial-time Turing reduction* M_i *then for almost all* x *there exists a* y *in* $Q(M_i, A, <i,x>) \cap A$ *such that* $|y| > |<i,x>|$.

Proof: Let $\{M_i\}_{i \in I\!N}$ be an enumeration of logarithmic-space bounded oracle machines. We define L_A as follows:

$$<i,x> \in L_A \Leftrightarrow \text{ the simulation of } M_i^{A^{\leq |<i,x>|}} \text{ uses } \leq S(|<i,x>|) \text{ tape cells and } <i,x> \notin L(M_i, A^{\leq |<i,x>|}).$$

CLAIM 7.1 $L_A \in NSPACE(S(n))$.

Proof: Since $A \in NSPACE(S(n))$, there exists a nondeterministic $S(n)$-space bounded Turing machine M_A, which accepts A. Immerman [8] and Szelepcsény [12] showed independently that nondeterministic space is closed under complementation. Therefore the complement of A ($\overline{A}$) is also in $NSPACE(S(n))$, and is recognized by a nondeterministic $S(n)$-space bounded Turing machine $M_{\overline{A}}$. We are first going to construct a machine that recognizes $\overline{L_A}$. Note that $<i,x> \in \overline{L_A}$ iff simulation of M_i uses more than $S(|<i,x>|)$ tape cells or $<i,x> \in L(M_i, A^{\leq|<i,x>|})$. Consider the following machine M :

```
input <i,x>
    mark off S(|<i,x>|) tape cells.
    Simulate M_i on input x
    if M_i queries a then
      if |a| > |<i,x>| continue computation of M_i in the NO state¹
      else guess if a is in A or in Ā and run M_A or M_Ā on input a.
      if it rejects then REJECT
        else if M_A accepts then continue M_i in the YES state
        else if M_Ā accepts then continue M_i in the NO state.
    ACCEPT iff M_i accepts or simulation of M_i
    uses more then S(|<i,x>|) tape cells.
end.
```

It is easy to see that $L(M)$ is in $NSPACE(S(n))$. Hence the complement of $L(M)$ is also in $NSPACE(S(n))$, by some machine $\overline{M}$. So $<i,x> \notin L(M)$ iff $<i,x> \in L(\overline{M})$ iff $<i,x> \in L_A$. This shows that L_A is in $NSPACE(S(n))$. ⊠

CLAIM 7.2 *If $L_A \leq_T^{logspace} A$ via M, then for all but finitely many x, $Q(M, A, <i,x>) \cap A^{>|<i,x>|} \neq \emptyset$.*

Proof: Suppose that $L_A \leq_T^{logspace} A$ and that M_j is a logarithmic-space bounded oracle machine such that $L_A = L(M_j, A)$. Let x be a string such that the simulation of M_j uses $\leq S(|<j,x>|)$ tape cells. (Since $S(n)$ majorizes $LOGSPACE$ this is true for sufficiently long x.) Then there exist at least one y in $Q(M_j, A, <j,x>) \cap A$ such that $|y| > |x|$.

Suppose otherwise. That is, the length of each element of $Q(M_j, A, <j,x>) \cap A$ is $\leq |x|$. Then $M_j^{A^{\leq|<j,x>|}}(<j,x>) = M_j^A(<j,x>)$, and $<j,x> \in L(M_j, A^{\leq|<j,x>|})$ iff $<j,x> \in L(M_j, A)$. Thus $<j,x> \in L_A$ iff $<j,x> \notin L(M_j, A)$. Which contradicts the fact that $L_A = L(M_j, A)$. ⊠

⊠

Note that contrary to section 3 we now have not shown that $NSPACE((S(n))$ has complete sets under honest reductions. Just that for every complete set there is some set reducing l.i. to it (in the weak sense above). However we can transform this notion to a property of complete sets against which we can diagonalize as stated in:

Corollary 4 *Every $\leq_T^{logspace}$ -complete set in $NSPACE(S(n))$, has an FT-subset.*

---

[1] To check if $|a| > |<i,x>|$, we use a counter keeping track of the number of symbols written on the oracle tape between two queries

Proof: Let A be a $\leq_T^{logspace}$ -complete set in $NSPACE(S(n))$. We now construct the set L_A w.r.t. A in the same way as in theorem 1. Since A is $\leq_T^{logspace}$ -complete for $NSPACE(S(n))$, we can now apply theorem 7: For almost all x there exists a y_x in $Q(M_i, A, <i, x>) \cap A$ such that $|y_x| > |x|$. We now define the following function g :

$$g(x) = \begin{cases} y_x & \text{if } y_x \text{ exists} \\ \text{some element of } A & \text{otherwise} \end{cases}$$

The function g is total and length increasing because for almost all $x \in \Sigma^*$, $|g(x)| > |x|$, and $g(x) \in A$. Furthermore $g \in FT$, which proofs the corollary. ⊠

This property of $\leq_T^{logspace}$ complete sets is easily translated to the other completeness notions.

Corollary 5 *Let $A \subseteq \Sigma^*$.*

1. *If A is $\leq_{tt}^{logspace}$-complete for $NSPACE(S(n))$, then A has an Ftt-subset.*
2. *If A is $\leq_{btt}^{logspace}$-complete for $NSPACE(S(n))$, then A has an Fbtt-subset.*
3. *If A is $\leq_m^{logspace}$-complete for $NSPACE(S(n))$, then A has an Fm-subset.*

Proof: The proof is similar to the proof of Corollary 4 and is left to the reader. ⊠

On the way to the differentiation between completeness notions we pick up two results on many-one complete sets.

Corollary 6 *No $\leq_m^{logspace}$-complete set for $NSPACE(S(n))$ is LOGSPACE-immune.*

Proof: Let C be any $\leq_m^{logspace}$-complete set for $NSPACE(S(n))$. Consider the set L_C and a many-one reduction from L_C to C via machine M_c. Applying theorem 7 it follows that for almost all x M_c queries a y to C such that $|y| > |x|$ and $y \in C$. So the set $\{Q(M_c, <i, 0^n>) |\ n > n_0\}$ for some n_0 large enough, is an infinite subset of C. Consider the following machine M, which accepts this subset:

input x
  $n := |x|$
  for all n', $n_0 < n' < n$ **do**
    run M_c on input $<c, 0^{n'}>$
    if $Q(M_c, <c, 0^{n'}>) = x$ **then** ACCEPT
  end do
  REJECT

Since M_c is a logarithmic-space bounded oracle machine it is easy to see that M is also a logarithmic-space bounded machine. Furthermore M accepts only if $x \in C$ and $||L(M)|| = \aleph_0$. ⊠

Corollary 7 *Every $\leq_m^{logspace}$-complete set for $NSPACE(S(n))$ has infinitely many disjoint, infinite subsets $\{B_i\}_{i \in \mathbb{N}}$, which are in LOGSPACE.*

Proof: Corollary 6 states that every $\leq_m^{logspace}$-complete set has an infinite subset in *LOGSPACE*. Let A be a $\leq_m^{logspace}$-complete set for $NSPACE(S(n))$ and let B_0 be such an infinite subset. Consider the set $A_1 = A \backslash B_0$. A_1 is $\in NSPACE(S(n))$ and A_1 is $\leq_m^{logspace}$-complete via the following reduction from A to A_1:

> input x
>   **if** $x \in B_0$ **then** output a fixed $y \notin A_1$
>   **else** output x
> **end**

Now we can apply corollary 6 again on A_1. This process can be repeated infinitely often and will generate the subsets $\{B_i\}_{i \in \mathbb{N}}$ as promised.⊠

4.2 Differences between complete sets in NSPACE

To this point we have obtained some useful properties of $NSPACE(S(n))$-complete sets. We now turn to differentiating between the various completeness notions for these space classes. We have shown that every $\leq_m^{logspace}$-complete set in $NSPACE(S(n))$ has an Fm subset. Hence to construct a $\leq_{2-d}^{logspace}$-complete set, which is not $\leq_m^{logspace}$-complete, it is sufficient to construct a set, which is $\leq_{2-d}^{logspace}$-complete but has no Fm subsets. For the construction we make use of a set K_S which is the space analog of the set K used in section 3 and can easily be seen to be many-one logspace complete for $NSPACE(S(n))$ (See [6])

Theorem 8 *There exists a $\leq_{2-d}^{logspace}$-complete set D in $NSPACE(S(n))$, which is not $\leq_m^{logspace}$-complete.*

Proof: The construction is essentially the same as the construction of Theorem 3. Again D will be constructed so that its only elements are of the form $<e, x, l, i>, i \in \{0, 1\}$. Then D will also be complete via the logspace reduction:

$$<e, x, l> \in K_S \leftrightarrow [<e, x, l, 0> \in D] \vee [<e, x, l, 1> \in D]$$

To ensure that D is not $\leq_m^{logspace}$-complete we diagonalize against all possible $\leq_m^{logspace}$-reductions from $\Sigma^*$to D Now let f_i be the i^{th} logspace computable function in some fixed enumeration of all such functions. As observed above we may not assume that the computation of f_i can be done in space $S(n)$. Therefore we diagonalize not against f_n at stage n of the diagonalization but against $f_{\pi_1(n)}$ the projection of the pair $n = <i, j>$ on its first coordinate. This means that we will encounter f_i an infinite number of times in the diagonalization with ever larger strings as argument. As of a certain length the simulation will (continue to) succeed within $S(n)$ space. Hence we create for each f_i an infinite sequence of counterexamples to the statement of Corollary 5 which says that the range of some function must be almost entirely within D if D is to be $\leq_m^{logspace}$ complete.

As we diagonalize against a length increasing property the sequence of numbers $\{b_n\}_n$, $b_0 = 0$ to use as boundaries can easily be computed by the diagonalization. We initially set $D = \emptyset$. Assume we have constructed D through stage $n - 1$.

stage n:

If $f_{\pi_1(n)}(0^{b(n-1)})$ can be computed within $S(b(n-1))$ tape cells
then let $y = f_i(0^n)$ otherwise $y = 0$.
For all s of the form $<e, x, l, i>$, $s \neq y$,
put $s \in D \leftrightarrow <e, x, l> \in K_S$; set $b_n = \max\{b(n-1)+1, |y|\}$

end of stage n

CLAIM 8.1 $D \in NSPACE(S(n))$

Proof: On input $s = <e, x, l, i>$ first check that $<e, x, l> \in K_S$. This can be done in $S(|<e, x, l>|)$-space. If this test succeeds, the construction can be carried out until a stage n is reached where $b(n) \geq |s|$. Note that since only $|s|$ bits of $b(n)$ have to be computed, this computation can be carried out in $\log(|s|)$ space. Now we can accept the input iff it is not explicitly put into $\overline{D}$ by the diagonalization, which can only occur if $b(n) = |s|$ and $f_{\pi_1(n)}(0^{b(n-1)})$ can be computed in $S(b(n-1)) \leq S(b(n))$ tape cells, and $f_{\pi_1(n)}(0^{b(n-1)}) = s$. These conditions can be checked in $S(|s|)$-space.⊠

CLAIM 8.2 *D is not $\leq_m^{logspace}$ complete.*

Proof: If D were $\leq_m^{logspace}$ complete, then there would be, according to Corollary 5, a logspace computable function f_i such that for almost all x : $f_i(x) \in D$. Take n large enough s.t.:

1. $f_i(x)$ can be computed in space $S(|x|)$ for all x with $|x| \geq b(n-1)$
2. $f_i(x) \in D$ for all $|x|$ with $|x| \geq n$ and with
3. $\pi_1(n) = i$.

Now consider stage n and find that $f_i(0^{b(n-1)}) \notin D$ which gives a contradiction ⊠

⊠

As a $\leq_{2-d}^{logspace}$-complete set is also $\leq_{k-d}^{logspace}$-complete, we immediately have,

Corollary 8 *1. For any integer $k > 1$, there exists a $\leq_{k-d}^{logspace}$-complete set which is not $\leq_m^{logspace}$-complete for $NSPACE(S(n))$*

2. For any integer $k > 1$, there exists a $\leq_{k-tt}^{logspace}$-complete set which is not $\leq_m^{logspace}$-complete for $NSPACE(S(n))$

Using the same technique we can construct a set which is $\leq_d^{logspace}$-complete but not $\leq_{btt}^{logspace}$-complete. We make use of the fact that any btt-reduction can only ask a bounded number of strings of the oracle set. On the other hand we can make a disjunctive tt-reduction dependent on a *growing* number of strings.

Theorem 9 *There exists a $\leq_d^{logspace}$-complete set D in $NSPACE(S(n))$, which is not $\leq_{btt}^{logspace}$-complete*

Proof: Let bin(i) = the binary representation of i, and $c(i,x) = 0^m\text{bin}(i)x$ where, for $x \in \Sigma^*$, $1 \le i \le |x|$ we have $|0^m\text{bin}(i)x| = 2|x|$. ¿From K_S we define the sets K_S^x as $K_S^x = \{c(i,x) | 1 \le i \le |x| \text{ and } x \in K_S\}$. For every x in K_S, we put at least one element of $K_S^x \in D$ will then be complete via the $\le_d^{logspace}$-reduction:

$$<e,x,l> \in K_S \leftrightarrow \exists i \le |x| \text{ such that } c(i,x) \in D$$

.

On the other hand we must ensure that, for every length increasing function f in $Fbtt$, $f(x)$ is not in D, for almost all x. Initially D is $\emptyset$, and $b(0) = 0$ Assume we have constructed D through stage $n-1$. Let M_i be the transducer computing f_i.

stage n:

If $Q(M_i, 0^{b(n-1)})$ can not be computed using less than $S(n)$ tape cells then $b(n) = b(n-1)+2$, and we put into D any $c(0,x)$, where $b(n-1) < 2|x| \le b(n)$ and $x \in K_S$. If $Q(M_i, 0^{b(n-1)})$ can be computed using less than $S(n)$ tape cells then any string in this set of length $\ge b(n-1)$ will not be in D.
$b(n) = \max\{|y| | y \in Q(M_i, 0^{b(n-1)})\}$. Therefore let z be any string in K_S^x for $b(n-1) < 2|x| \le b(n)$. Put z in D iff $z \notin Q(M_i, 0^b(n-1))$

end of stage n

CLAIM 9.1 $D \in NSPACE(S(n))$.

Proof: As before, on input $s = 0^m\text{bin}(i)x$ we first check that $x \in K_S$. Next n can be computed by simulating stages until $b(n) > 2|x|$ in logarithmic space. Note that while we are computing this number we are only interested in the length of the longest element in a set of queries if this set can be generated within space $S(m)$ bounded for $m \le b(n) \le 2|x|$. When we have computed this number we know the index of the function to simulate.

If the associated set of this function cannot be computed within space bounded by $S(b(n-1))$ where $b(n-1) \le 2|x|$then accept; otherwise reject only if $s \in Q(M_i, 0^{b(n-1)})$. As $b(n-1) < 2|x|$ this question can be settled in $S(b(n-1))$ space so it can certainly be settled in $S(2|x|)$ space. (Note that we don't need to store any of the strings in $Q(M_i, 0^{b(n-1)})$, whence this algorithm also works for sublinear space.) ⊠

CLAIM 9.2 *D is not $\le_{btt}^{logspace}$-complete.*

Proof: If D were $\le_{btt}^{logspace}$-complete, then there would be, according to Corollary 5, an Fbtt subset in D generated by some Fbtt function g_i, corresponding to a function f_i in such a way that for all x, $g_i(x) \in Q(M_i, x)$. Now take n (large enough) such that:

1. $f_i(x)$ can be computed in $NSPACE(S(n))$ for all x with $|x| \ge b(n-1)$
2. $g_i(x) \in D$ for all $|x|$ with $|x| \ge n$ and
3. $\pi_1(n) = i$.

Now at stage n, $g_i(0^{b(n-1)})$ is an element of $Q(M_i, 0^{b(n-1)})$ which has length $> b(n-1)$ and is therefore not in D which gives a contradiction. ⊠

⊠

As disjunctive truth table completeness implies truth table completeness this implies.

Corollary 9 *There exists a $\leq_{tt}^{logspace}$-complete set which is not $\leq_{btt}^{logspace}$-complete*

5 Conclusions and Further Work

The question of whether $\leq_{tt}^{logspace}$-complete sets can be separated from $\leq_{T}^{logspace}$-complete sets for the nondeterministic space classes remains open. The proof that achieves this separations for nondeterministic time classes seemingly does not work for this case.

One might also consider more efficient reductions such as those (for example, NC-reductions) which have been proposed to compare the parallel complexity of problems. The methods given here might lead to differences between completeness notions for the various parallel complexity classes.

One area of great interest would be to separate the various polynomial time reductions on the classes between P and PSPACE, and in particular to do this for NP. Of course, such results would imply inequalities between these classes and so are not tractable at this point in time. Even assuming such separations, these results are not currently known. For example, even assuming P different from NP (or P different from PSPACE) there is no known difference between $\leq_m^p$ and $\leq_T^p$ for sets in NP (PSPACE). Furthermore, no structural properties of sets in these classes is known to imply differences between these reductions. The techniques discussed here may point the way toward the first examples of such results.

Bibliography

[1] Balcázar J.L., J. Díaz & J. Gabarró. *Structural Complexity I.* W. Brauer, G. Rozenberg & A. Salomaa (eds.) EATCS Monographs on Theoretical Computer Science 11 (1988) Springer Verlag.

[2] Berman L. *On the structure of complete sets.* Proc. 17th IEEE conference on Foundations of Computer Science (1976) pp76-80.

[3] Berman, L. & J. Hartmanis. *On isomorphisms and density of NP and other complete sets.* SIAM J. on Computing 1 (1977) pp. 305–322.

[4] Cook, S. A. *The complexity of theorem-proving procedures.* Proc. 3d ACM Symp. on Theory of Computing, Assoc. for Computing Machinery, New York (1971) pp. 151–158.

[5] Ganesan, K & S. Homer. *Complete problems and strong polynomial reducibilities.* Boston University Technical Report ♯88-001, January, 1988. Aspects of Computer Science, Springer Lecture Notes in Computer Science 349 (1989) pp. 240–250.

[6] Hartmanis, J. *Feasible Computations and Provable Complexity Properties.* NSF regional conference series in applied mathematics (1978).

[7] Hartmanis, J. *On the logtape isomorphism of complete sets* Theoretical Computer Science 7 (1978) pp. 273–286.

[8] Immerman, N. *Nondeterministic space is closed under complementation.* SIAM J. on Computing 17 (1988) pp. 935–938.

[9] Karp, R.M. *Reducibility among combinatorial problems.* Complexity of Computer Computations, R.E. Miller & J.W. Thatcher eds. Plenum N.Y. pp. 85–103.

[10] Jones, N. *Space bounded reducibilities among combinatorial problems* J. Comp. System Sci. 11 (1975) pp. 68–85.

[11] Ladner, R.E., N. Lynch & A.L. Selman. *A comparison of polynomial time reducibilities.* Theoretical Computer Science 1 (1975) pp. 103–123.

[12] Szelepcsényi, R. *The method of forcing for nondeterministic automata.* Bulletin of the EATCS 33 (1987) pp. 96–100.

[13] Watanabe, O. *A comparison of polynomial time completeness notions.* Theoretical Computer Science 54 (1987) 249–265.

[14] Watanabe O. *On the Structure of Intractable Complexity Classes*, Ph.D. thesis dept. of Computer Science, Tokyo Institute of Technology 1987.

The ITLI Prepublication Series

1986

86-01		The Institute of Language, Logic and Information
86-02	Peter van Emde Boas	A Semantical Model for Integration and Modularization of Rules
86-03	Johan van Benthem	Categorial Grammar and Lambda Calculus
86-04	Reinhard Muskens	A Relational Formulation of the Theory of Types
86-05	Kenneth A. Bowen, Dick de Jongh	Some Complete Logics for Branched Time, Part I Well-founded Time, Forward looking Operators
86-06	Johan van Benthem	Logical Syntax

1987

87-01	Jeroen Groenendijk, Martin Stokhof	Type shifting Rules and the Semantics of Interrogatives
87-02	Renate Bartsch	Frame Representations and Discourse Representations
87-03	Jan Willem Klop, Roel de Vrijer	Unique Normal Forms for Lambda Calculus with Surjective Pairing
87-04	Johan van Benthem	Polyadic quantifiers
87-05	Víctor Sánchez Valencia	Traditional Logicians and de Morgan's Example
87-06	Eleonore Oversteegen	Temporal Adverbials in the Two Track Theory of Time
87-07	Johan van Benthem	Categorial Grammar and Type Theory
87-08	Renate Bartsch	The Construction of Properties under Perspectives
87-09	Herman Hendriks	Type Change in Semantics: The Scope of Quantification and Coordination

1988

Logic, Semantics and Philosophy of Language:

LP-88-01	Michiel van Lambalgen	Algorithmic Information Theory
LP-88-02	Yde Venema	Expressiveness and Completeness of an Interval Tense Logic
LP-88-03		Year Report 1987
LP-88-04	Reinhard Muskens	Going partial in Montague Grammar
LP-88-05	Johan van Benthem	Logical Constants across Varying Types
LP-88-06	Johan van Benthem	Semantic Parallels in Natural Language and Computation
LP-88-07	Renate Bartsch	Tenses, Aspects, and their Scopes in Discourse
LP-88-08	Jeroen Groenendijk, Martin Stokhof	Context and Information in Dynamic Semantics
LP-88-09	Theo M.V. Janssen	A mathematical model for the CAT framework of Eurotra
LP-88-10	Anneke Kleppe	A Blissymbolics Translation Program

Mathematical Logic and Foundations:

ML-88-01	Jaap van Oosten	Lifschitz' Realizabiility
ML-88-02	M.D.G. Swaen	The Arithmetical Fragment of Martin Löf's Type Theories with weak Σ-elimination
ML-88-03	Dick de Jongh, Frank Veltman	Provability Logics for Relative Interpretability
ML-88-04	A.S. Troelstra	On the Early History of Intuitionistic Logic
ML-88-05	A.S. Troelstra	Remarks on Intuitionism and the Philosophy of Mathematics

Computation and Complexity Theory:

CT-88-01	Ming Li, Paul M.B.Vitanyi	Two Decades of Applied Kolmogorov Complexity
CT-88-02	Michiel H.M. Smid	General Lower Bounds for the Partitioning of Range Trees
CT-88-03	Michiel H.M. Smid, Mark H. Overmars, Leen Torenvliet, Peter van Emde Boas	Maintaining Multiple Representations of Dynamic Data Structures
CT-88-04	Dick de Jongh, Lex Hendriks, Gerard R. Renardel de Lavalette	Computations in Fragments of Intuitionistic Propositional Logic
CT-88-05	Peter van Emde Boas	Machine Models and Simulations (revised version)
CT-88-06	Michiel H.M. Smid	A Data Structure for the Union-find Problem having good Single-Operation Complexity
CT-88-07	Johan van Benthem	Time, Logic and Computation
CT-88-08	Michiel H.M. Smid, Mark H. Overmars, Leen Torenvliet, Peter van Emde Boas	Multiple Representations of Dynamic Data Structures
CT-88-09	Theo M.V. Janssen	Towards a Universal Parsing Algorithm for Functional Grammar
CT-88-10	Edith Spaan, Leen Torenvliet, Peter van Emde Boas	Nondeterminism, Fairness and a Fundamental Analogy
CT-88-11	Sieger van Denneheuvel, Peter van Emde Boas	Towards implementing RL

Other prepublications:

X-88-01	Marc Jumelet	On Solovay's Completeness Theorem

1989

Logic, Semantics and Philosophy of Language:

LP-89-01	Johan van Benthem	The Fine-Structure of Categorial Semantics
LP-89-02	Jeroen Groenendijk, Martin Stokhof	Dynamic Predicate Logic, towards a compositional, non-representational semantics of discourse
LP-89-03	Yde Venema	Two-dimensional Modal Logics for Relation Algebras and Temporal Logic of Intervals
LP-89-04	Johan van Benthem	Language in Action
LP-89-05	Johan van Benthem	Modal Logic as a Theory of Information
LP-89-06	Andreja Prijatelj	Intensional Lambek Calculi: Theory and Application
LP-89-07	Heinrich Wansing	The Adequacy Problem for Sequential Propositional Logic

Mathematical Logic and Foundations:

ML-89-01	Dick de Jongh, Albert Visser	Explicit Fixed Points for Interpretability Logic
ML-89-02	Roel de Vrijer	Extending the Lambda Calculus with Surjective Pairing is conservative
ML-89-03	Dick de Jongh, Franco Montagna	Rosser Orderings and Free Variables
ML-89-04	Dick de Jongh, Marc Jumelet, Franco Montagna	On the Proof of Solovay's Theorem
ML-89-05	Rineke Verbrugge	Σ-completeness and Bounded Arithmetic
ML-89-06	Michiel van Lambalgen	The Axiomatization of Randomness
ML-89-07	Dirk Roorda	Elementary Inductive Definitions in HA: from Strictly Positive towards Monotone
ML-89-08	Dirk Roorda	Investigations into Classical Linear Logic

Computation and Complexity Theory:

CT-89-01	Michiel H.M. Smid	Dynamic Deferred Data Structures
CT-89-02	Peter van Emde Boas	Machine Models and Simulations
CT-89-03	Ming Li, Herman Neuféglise, Leen Torenvliet, Peter van Emde Boas	On Space efficient Simulations
CT-89-04	Harry Buhrman, Leen Torenvliet	A Comparison of Reductions on Nondeterministic Space
CT-89-05	Pieter H. Hartel, Michiel H.M. Smid, Leen Torenvliet, Willem G. Vree	A Parallel Functional Implementation of Range Queries
CT-89-06	H.W. Lenstra, Jr.	Finding Isomorphisms between Finite Fields
CT-89-07	Ming Li, Paul M.B. Vitanyi	A Theory of Learning Simple Concepts under Simple Distributions and Average Case Complexity for the Universal Distribution (Prel. Version)
CT-89-08	Harry Buhrman, Steven Homer, Leen Torenvliet	Honest Reductions, Completeness and Nondeterminstic Complexity Classes

Other Prepublications:

X-89-01	Marianne Kalsbeek	An Orey Sentence for Predicative Arithmetic
X-89-02	G. Wagemakers	New Foundations: a Survey of Quine's Set Theory
X-89-03	A.S. Troelstra	Index of the Heyting Nachlass
X-89-04	Jeroen Groenendijk, Martin Stokhof	Dynamic Montague Grammar, a first sketch
X-89-05	Maarten de Rijke	The Modal Theory of Inequality