

Non-Uniform Reductions

Harry Buhrman¹, Benjamin Hescott², Steven Homer³, and Leen Torenvliet⁴

¹ CWI; Kruislaan 409; 1098 SJ Amsterdam; the Netherlands

² Tufts University; Computer Science Dpt.; 161 College Ave; Medford, MA 02155

³ Boston University; Computer Science Dpt.; 111 Cummington St; Boston, MA 02215

⁴ ILLC; Plantage Muidergracht 24; 1018 TV Amsterdam; the Netherlands

Abstract. We study properties of non-uniform reductions and related completeness notions. We strengthen several results of Hitchcock and Pavan [HP06] and give a trade-off between the amount of advice needed for a reduction and its honesty on NEXP. We construct an oracle relative to which this trade-off is optimal. We show, in a more systematic study of non-uniform reductions, among other things that non-uniformity can be removed at the cost of more queries. In line with Post’s program for complexity theory [BT05] we connect such ‘uniformization’ properties to the separation of complexity classes.

keywords: Complexity classes, non-uniform complexity, reductions, non-relativizing methods.

1 Introduction

Reductions and completeness are two of the original concepts in complexity theory and form part of the core of the field to this day. Determining whether a problem is complete for a particular class is central in determining the computational complexity of that problem. In a broader perspective, questions of whether different types of reductions are the same or different, and whether completeness notions induced by these reductions are the same or different have been assiduously explored, e.g., [LLS75,BHT91,Wat87,BST93a]. Answers to these questions are related to the central open problems of the field. In some cases, inequality of reductions on certain complexity classes implies inequality of complexity classes [LM94]. In others, the collapse of degrees to an isomorphism type under some notion of reduction also yields the inequality of classes [BH77,FFK92,HH91,HS92,KMR89,You90,AB93,All88]. This makes the properties of reductions, for example whether they are length-increasing, 1-1, etc. important objects of study. Finally, certain properties of complete sets in different complexity classes might have implications for the question of equality of these classes [AS84,BvMFT00,BT04].

The lion’s share of investigations of reductions has been into *uniform* reductions. Now, non-uniformity has entered the realm of the reduction. As with the definitions of non-uniform complexity classes, by means of advice classes, (P/poly and families of circuits of a certain size), one can define reductions that are computable by means of additional advice or by polynomial size circuits.

Allender et. al. [ABK⁺02] have shown that the set R of Kolmogorov incompressible strings, with respect to exponential time Kolmogorov complexity, is complete for EXP with respect to polynomial time (truth-table) reductions that have a polynomial amount of *advice*. Moreover, the advice is indispensable: R is not complete with respect to *uniform* Turing reductions [BM95]. For a strengthening of this see the thesis of Ronneburger [Ron04].

Agrawal [Agr02], while studying the isomorphism conjecture [BH77] for NP complete sets, used non-uniform reductions with advice. He showed under the assumption that a certain type of one-way function exists, that all many-one complete sets for NP are 1-1 and length increasing complete for reductions that use some amount of non-uniform advice.

Recently Hitchcock and Pavan [HP06] showed under a different assumption, namely that NP is not small (does not have resource bounded measure zero), that every many-one complete set for NP is length increasing complete with reductions that use a polynomial amount of advice. Moreover they also show that for NEXP the many-one complete sets are length increasing complete with reductions that use a polynomial amount of advice.

In this paper we improve the results of Hitchcock and Pavan. In particular, we show, under the weaker assumption that there exists a $\text{DTIME}(2^{n^c})$ bi-immune set in NP, that one bit of advice suffices to show that many-one complete sets are length increasing complete⁵. We also improve their results for NEXP. We again reduce the number of advice bits needed to make the reductions length increasing and show that our result is optimal relative to some oracle. In particular, this yields an oracle where the many-one complete sets for NEXP are not (uniformly) length increasing complete, in fact they are only exponentially honest, matching the best known result due to Ganesan and Homer [GH88]. Surprisingly, this shows that non-relativizing techniques are needed to settle this question.

Another structural property known for EXP [HKR93] and for NEXP [BST93b] is the following. Every 1-truth-table complete set is many-one complete. We show that under the hypothesis that NP contains a set that is $\text{NTIME}(2^{n^c}) \cap \text{co-NTIME}(2^{n^c})$ bi-immune every 1-truth-table complete set for NP is many-one complete with 1 bit of advice. We also construct, extending an earlier result of [BT99], an oracle world where such a bi-immune set exists in NP.

The results above warrant a more systematic investigation into non-uniform reductions. In particular, is the amount of advice needed for the non-uniform reductions above optimal? In general, when does advice yield additional power?

In the second part of this paper we begin such a study. We first show that for EXP constant query reductions that have advice are strictly more powerful than their uniform counterparts. For example we show that the $\leq_m^{p/1}$ -complete degree and the \leq_{tt}^p -complete degree for EXP are incomparable. Analogous to uniform reductions, we show that complete sets for EXP with respect to many-one reductions that use c -bits of advice are 1-1 and length increasing complete

⁵ Technically we show something slightly weaker. We show that the many-one reduction is either length increasing or accepts/rejects without querying a string. The result does hold for 1-tt reductions.

with c -bits of advice. These results require new techniques as the original ideas for uniform reductions cannot be used directly.

Second, we show that non-uniformity can be removed at the price of more queries. In particular we show that sets that are many-one complete with respect to reductions that use $O(\log n)$ bits of advice are Turing-complete with respect to uniform reductions for EXP. For the delta levels of the polynomial hierarchy we show that truth-table reductions suffice in order to ‘uniformize’ the reduction. Such results can be seen as a Karp-Lipton [KL80] type result for reductions. Here we use ideas and techniques from the paper that studies the auto-reducibility of complete sets [BvMFT00]. We have the added bonus that these theorems do not relativize and hope that these results could be used in a nonrelativizing proof that separates complexity classes. In particular it follows from our results that solving the question of whether many-one complete sets with 1 bit of advice are uniformly truth-table complete will separate complexity classes (EXP from EXPSPACE or PH from EXP). Hence understanding questions like these has deep implications for complexity theory.

2 Notation

We adopt the standard definitions and notations for well-known complexity classes, and other notions of computational complexity, as can be found, e.g., in [BDG88], [HS01] and [Pap94]. We recall the notion of the *extended* many-one reduction: $A \leq_m^p B$ if there exists a polynomial time Turing machine M with an output tape, where on input x , M does one of the following: M outputs $f(x) \in \Sigma^*$ and $x \in A \leftrightarrow f(x) \in B$, M outputs ACCEPT and $x \in A$, or M outputs REJECT and $x \notin A$.

Polynomial time oracle machines are used to characterize both Turing and truth-table reductions. The *query set* of an oracle machine M on input x with oracle A is denoted as $Q(M^A(x))$. The notation $M^A(x)$ is also used as a notation for the outcome of the computation of machine M on input x with oracle A . This can be either accept/reject or a string y . In the latter case—the machine computes a function—we also use the notation $f^A(x)$. We assume enumerations $\{M_i\}_i$ ($\{f_i\}_i$) for all convenient classes of machines (functions). A polynomial reduction is nonuniform if it is in $P/poly$, i.e., $A \leq_m^{P/poly} B$ if $\exists f \in PF$ and $h \in poly$ where $A = \{x \mid f(x, h(|x|)) \in B\}$. Here $poly = \{g \mid \forall n, |g(n)| \leq p(n) \text{ for some polynomial } p\}$. Many times the amount of advice is more or less restrictive, this restriction is on the function h in the definition above. Namely, for one bit of advice the range of h is $\{0, 1\}$. We denote a one-bit nonuniform many one reduction, $\leq_m^{p/1}$. In many complexity classes K will stand for the generic complete set, e.g., $K^A = \{\langle i, x, k \rangle \mid M^A(x) \text{ accepts in } k \text{ steps}\}$ is the canonical complete set for EXP with oracle A . To avoid confusion we will subscript K with the appropriate complexity class, e.g., K_{EXP} . M_K or N_K will be the machine accepting this set for deterministic, or resp. nondeterministic complexity classes.

In this work we consider the measure hypothesis on NP formulated by Lutz which states that NP does not have p -measure 0, $\mu(\text{NP}) \neq 0$. This is equivalent to assuming that there is no polynomial time martingale that succeeds on every language in NP. The measure hypothesis on NP also implies the existence of a $\text{DTIME}(2^{n^\epsilon})$ -bi-immune set within NP [HP04].

A set is C -immune if it does not have any infinite subsets in C , it is C -bi-immune if it and its complement are both C -immune.

3 Advice to Strengthen Reductions

3.1 Length increasing reductions

Hitchcock and Pavan [HP06] recently showed that some reductions can be made length increasing under the assumption that NP is not small. We improve upon the advice needed in their paper in the following theorem.

Theorem 1. $\mu(\text{NP}) \neq 0 \implies \text{Every } \leq_m^p\text{-complete set is } \leq_{m,\text{li}}^{p/O(\log \log n)}\text{-complete.}$

Proof. Choose $\epsilon > 0$ and let R be a $n^{1+\epsilon}$ -random set in NP. For every n let $x_0^n, \dots, x_{2^{\log n+2}}^n$ be the lexicographically first $2^{\log n+2}$ strings of length n . We claim that $(\forall n)(\exists i \leq \log n + 1)[R(0^n) \oplus R(x_i) = 1]$. Suppose not, then the following betting strategy wins infinitely on R . Start with reserving $\frac{1}{2n^2}$ of the starting capital for use at length n . Now bet evenly on $\{0^n\}$ and use the outcome of $R(0^n)$ to bet on $x_1^n, \dots, x_{2^{\log n+2}}^n$. The ‘‘capital at length n ’’ is $2^{2^{\log n+2}}/2n^2 > 1$. To identify an index i for which $R(0^n) \oplus R(x_i) = 1$ we need about $\log \log n$ bits. Let

$$D = \{\langle x, y, \phi \rangle \mid |x| = |y| = |\phi| \wedge R(x) + R(y) + \text{SAT}(\phi) \geq 2\}.$$

Now let A be a \leq_m^p -complete set in NP, and let M calculate the reduction from D to A .

Suppose that for every k there are infinitely many $\langle x, y, \phi \rangle$ such that $|M(\langle x, y, \phi \rangle)| < |\langle x, y, \phi \rangle|^{1/k}$. The set A is in NP so it is $\text{DTIME}(2^{n^c})$ -computable for some c . Hence for infinitely many $\langle x, y, \phi \rangle$ it is $\text{DTIME}(2^n)$ computable whether $\langle x, y, \phi \rangle$ is in D . This gives a betting strategy since $M(\langle x, y, \phi \rangle) \notin A$ implies $x \notin R$ or $y \notin R$ and $M(\langle x, y, \phi \rangle) \in A$ implies $x \in R$ or $y \in R$. Assume x is lexicographically less than or equal to y . Given the outcome of $M(\langle x, y, \phi \rangle) \in A$ use $1/3d$ of the capital on $x \in R$ and, if necessary, the remaining $2/3d$ of the capital on $y \in R$. In either case the capital grows to $4/3$. So $\text{SAT} \leq_{m,\text{li}}^{p/\log \log n} D \leq_{m,\text{honest}}^p A$ which proves the theorem.

With a weaker assumption but a less standard notion of reduction we arrive at an even stronger conclusion.

Theorem 2. $(\forall c > 0)[(\exists R \in \text{NP})[R \text{ is } \text{DTIME}(2^{n^c})\text{-bi-immune}] \implies [A \text{ is } \leq_m^p\text{-complete} \Rightarrow A \text{ is } \leq_{m,\text{li}}^{p/1}\text{-complete}]]$.

Proof. Suppose that A is in $\text{NTIME}(n^d)$. Let

$$D = \left\{ \begin{array}{l} \langle \phi, 0 \rangle : \phi \in \text{SAT} \vee 0^{|\phi|} \in R \\ \langle \phi, 1 \rangle : \phi \in \text{SAT} \wedge 0^{|\phi|} \in R \end{array} \right\}$$

It is easy to see that $\text{SAT} \leq_{m,\text{li}}^{p/1} D$. It therefore suffices to prove that $D \leq_{\hat{m},\text{honest}}^p A$. Let M be the reduction from D to A . By definition of D ,

$$\begin{aligned} \phi \notin \text{SAT} &\implies [0^{|\phi|} \in R \Leftrightarrow M(\langle \phi, 0 \rangle) \in A] \\ \phi \in \text{SAT} &\implies [0^{|\phi|} \in R \Leftrightarrow M(\langle \phi, 1 \rangle) \in A] \end{aligned}$$

If $(\forall k)(\exists^\infty \phi)[|M(\langle \phi, b \rangle)| < |\langle \phi, b \rangle|^{1/k}]$, then $(\forall k)(\exists^\infty \phi)[|M(\langle \phi, 0 \rangle)| < |\langle \phi, 0 \rangle|^{1/k}]$ or $(\forall k)(\exists^\infty \phi)[|M(\langle \phi, 1 \rangle)| < |\langle \phi, 1 \rangle|^{1/k}]$.

Then by the $\text{DTIME}(2^{n^c})$ -bi-immunity of R and the fact that $M(\langle \phi, b \rangle)$ in A is $\text{NTIME}(n^d)$ and hence $\text{DTIME}(2^{n^d})$ computable if $|M(\langle \phi, b \rangle)| < |\langle \phi, b \rangle|^{1/k}$ we can conclude for almost all of these ϕ in the first case that $\phi \in \text{SAT}$; if $\phi \notin \text{SAT}$, we could calculate $0^{|\phi|} \in R$ in $\text{DTIME}(2^n)$. Similarly, in the second case we know that $\phi \notin \text{SAT}$. So, for a suitable k whenever $|M(\langle \phi, b \rangle)| < |\langle \phi, b \rangle|^{1/k}$, we can decide membership of ϕ in SAT in P . In the remaining cases M is honest.

Hitchcock and Pavan [HP06] showed that for nondeterministic exponential time, no assumption is needed to make the many-one complete sets length increasing complete via reductions that use a polynomial amount of advice. Next, we will improve this result to $n - \log n$ bits of advice.

Theorem 3. $(\forall c > 0)[A \leq_m^p - \text{complete for NEXP} \implies A \leq_{m,\text{li}}^{n-c \log n} \text{complete}]$

Proof. Let p be a 1-1 padding function for K_{NEXP} and let f be a 1-1 reduction from K_{NEXP} to A . $(\forall n)(\exists r)[|r| = n \wedge (\forall x)[|x| = n \implies |f(p(x, r))| > |x|]$. Given the first $|r| - c \log n$ bits of r and x , the length increasing non-uniform reduction f' tries $f(p(x, r.r'))$ for all r' of length $c \log n$ until it finds an output greater than $|x|$.

The same proof yields a trade-off between the amount of advice and the honesty of the reduction.

Theorem 4. *The many-one complete sets for NEXP are $g(|x|)$ -honest for reductions that use $g(|x|) - c \log n$ bits of advice.*

Corollary 1 ([GH88]). *For any c , the many-one complete sets for NEXP are $c \log n$ -honest complete.*

The amount of advice used in the previous theorem is optimal relative to some oracle.

Theorem 5. *There exists an oracle witnessing that NEXP has a many-one complete set that is not complete under length increasing reductions.*

Proof. We construct an oracle A , and sets C , and D so that C is complete in NEXP^A , but every length increasing reduction fails to reduce 0^* to C . The construction and the correctness proofs are quite involved and lengthy. Therefore, we move these to the appendix.

The construction can be adapted to reductions that have k bits of advice by the following observation. In stage i of the construction we diagonalize against many-one reduction M_i , resulting in the coding of n^i strings from K^A into strings of length $\log(n^i) = i \log n$. Instead of diagonalizing against just M_i the same construction can be used to diagonalize against 2^k different advices for M_i resulting in a coding of $2^k n^i$ strings from K^A into strings of length $\log(2^k n^i) = k + i \log n$. This exactly matches the bound from Theorem 4.

The oracle above works for nondeterministic polynomial time as well as non-deterministic exponential time. The only significant change in the constructions (see Appendix A.1) is that we pad the counter so that the replacement in C is only polynomially smaller in length than the original element taken from C . We omit the full proof here, but state the result as a theorem.

Theorem 6. *There exists an oracle witnessing that NP has a many one complete set that is not complete with length increasing reductions.*

3.2 1-truth-table versus many-one reductions

Sometimes 1-tt reductions can be converted into many-one reductions. Under a rather strong assumption we can prove this theorem for 1-tt reductions that use one bit of advice.

Theorem 7. *Let A be $\leq_{1\text{-tt}}^p$ complete for NP. If, for some c , there exists a set R in NP that is $\text{NTIME}(2^{cn}) \cap \text{co-NTIME}(2^{cn})$ -bi-immune, then A is also $\leq_m^{p/1}$ -complete for NP.*

Proof. Let A be $\leq_{1\text{-tt}}^p$ complete and let

$$D = \left\{ \begin{array}{l} \langle \phi, 0 \rangle : \phi \in \text{SAT} \vee 0^{|\phi|} \in R \\ \langle \phi, 1 \rangle : \phi \in \text{SAT} \wedge 0^{|\phi|} \in R \end{array} \right\}$$

Since $R \in \text{NP}$, $D \in \text{NP}$ and so $D \leq_{1\text{-tt}}^p A$. Obviously $\text{SAT} \leq_m^{p/1} D$. The single bit of advice needed is whether $0^{|\phi|} \in R$. Let M be the $\leq_{1\text{-tt}}^p$ reduction from D to A . On input $\langle \phi, 0 \rangle$ M produces a string z . Then, depending on the program of M , $\langle \phi, 0 \rangle \in D \leftrightarrow z \in A$ or $\langle \phi, 0 \rangle \in D \leftrightarrow z \notin A$. In the first case we say M is of type m (many-one) on input $\langle \phi, 0 \rangle$ and in the second case that M is of type \bar{m} on input $\langle \phi, 0 \rangle$. We now claim that the bi-immunity of R implies M can be of type \bar{m} for only finitely many unsatisfiable ϕ . Or, in other words,

$$\begin{aligned} (\exists^\infty \phi \notin \text{SAT})[\text{type}(M(\phi, 0)) = \bar{m}] \Rightarrow \\ \exists B, \|B\| = \infty, B \in \text{NTIME}(2^{2n}) \cap \text{co-NTIME}(2^{2n}), B \subseteq R \vee B \subseteq \bar{R}. \end{aligned}$$

We first prove this claim. Consider

$$C = \{0^n \mid \exists \phi \notin \text{SAT}, |\phi| = n, \text{type}(M(\phi, 0)) = \bar{m}\}.$$

Then $\|C\| = \infty \implies [\|C \cap R\| = \infty \vee \|C \cap \bar{R}\| = \infty]$. Assume $\|C \cap R\| = \infty$ —the case where $\|C \cap \bar{R}\| = \infty$ is analogous—and set $B = C \cap R$. Now $B \in \text{NTIME}(2^{2n})$ by the following algorithm. On input 0^n guess ϕ with $|\phi| = n$ and $\phi \notin \text{SAT}$; Check that $\text{type}(M(\phi, 0)) = \bar{m}$ and verify $0^n \in R$. It is also the case that \bar{B} is in $\text{NTIME}(2^{2n})$ by the following algorithm. On input 0^n check that for every ϕ of length n either $\phi \in \text{SAT}$ or that $\text{type}(M(\phi, 0)) = m$, or that there exists a $\phi \notin \text{SAT}$ of length n for which $\text{type}(M(\phi, 0)) = \bar{m}$ but $M(\phi, 0) \in A$ (which means $0^n \notin R$).

We conclude that under the assumption of the theorem there can only be finitely many unsatisfiable ϕ such that $M(\phi, 0)$ is of type \bar{m} . A similar proof shows that there can be only finitely many satisfiable ϕ such that $M(\phi, 1)$ is of type \bar{m} . From this we can build our many-one reduction.

The premise of the previous theorem seems rather strong. Yet it is not impossible that NP does have such sets. To provide evidence for this statement we construct precisely such a set in the following theorem.

Theorem 8. *For every constant c , there exists an oracle A such that NP has a set that is $\text{NTIME}(2^{n^c}) \cap \text{co-NTIME}(2^{n^c})$ -immune.*

Proof. We use a construction appearing in [BT99] in which the oracle is created from an infinite Kolmogorov random string. The construction is moved to the appendix due to space considerations.

4 When Does Advice Help?

As we stated in the introduction, reductions with advice seem to inhabit a higher level of complexity than do uniform reductions. In the next theorem we show a simple diagonalization that demonstrates incomparability of uniform 2-tt reductions and many-one reductions with advice.

Theorem 9. *There exists a set $A \in \text{EXP}$ that is $\leq_{2\text{-tt}}^p$ -hard for EXP but not $\leq_m^{p/1}$ -hard for EXP, and there exists a set $A' \in \text{EXP}$ that is $\leq_m^{p/1}$ -hard for EXP but not $\leq_{2\text{-tt}}^p$ -hard for EXP.*

Proof. First we show that there is a set $A \in \text{EXP}$ that is $\leq_{2\text{-tt}}^p$ -hard for EXP but not $\leq_m^{p/1}$ -hard for EXP. We start with an enumeration $\{M_i\}_i$ of all polynomial time transducers that take one bit of advice and look at the outcome of the computations on input 0^n for n spaced exponentially far apart. Our exponential-time computable set D that we use for diagonalization is a subset of $\{0\}^* \cup \{1\}^*$, where strings of this type appear with exponential gaps between them. Our complete set A is a subset of $\{\langle b, x \rangle \mid b \in \{0, 1\}\}$, and the 2-tt reduction from

K_{EXP} to A on input x computes $\langle 0, x \rangle \in A \oplus \langle 1, x \rangle \in A$ and accepts if the outcome is 1. On suitable inputs 0^n we diagonalize against M_i . If M_i accepts or rejects without querying, or queries a part of A that has already been set, i.e., exponentially smaller than the input, we diagonalize by letting $M_i(0^n)$ compute the wrong answer. If the queries are not of the form $\langle b, x \rangle$ then we do the same, so for the rest of the proof we assume that M_i on input 0^n computes a query that has not yet been set of the form $\langle b, x \rangle$, given advice 0 or 1. There are several possibilities:

1. M_i computes $\langle b_1, x \rangle$ with advice 0 and $\langle b_2, y \rangle$ with advice 1 where $x \neq y$. Then we set $0^n \in D$, $\langle b_1, x \rangle \notin A$, and $\langle b_2, y \rangle \notin A$, furthermore, set $\langle 1 - b_1, x \rangle$ in A if $x \in K$, and set $\langle 1 - b_2, y \rangle$ in A if $y \in K$.
2. M_i computes $\langle b, x \rangle$ both with advice 0 and 1. Then we set $0^n \in D$ and set $\langle 1 - b, x \rangle$ to reflect $x \in K$ correctly.
3. M_i computes $\langle b, x \rangle$ with advice 0 and $\langle 1 - b, x \rangle$ with advice 1. Now we set $0^n \in D$ and compute $M_i(1^n)$ with advice 0 and 1. The case of small query strings is treated in the same way, and the cases above are treated in the same way. If we cannot already diagonalize on 1^n alone then it must be the case that $M_i(1^n)$ computes $\langle b_y, y \rangle$ and $\langle 1 - b_y, y \rangle$. There are three cases.
 - (a) $x = y$ and $b = b_y$. In this case we set $0^n \in D$ and $1^n \notin D$. Set $\langle b, x \rangle$ and $\langle 1 - b, x \rangle$ consistent with $x \in K$.
 - (b) $b = 1 - b_y$. In this case we set $0^n \in D$, $1^n \in D$, and $M^0(0^n) = M^1(1^n) \notin A$.
 - (c) $x \neq y$. In this case we set $0^n \in D$ and $1^n \in D$ and let $M^0(0^n) \notin A$ and $M^1(1^n) \notin A$.

In the reverse case we have to deal with an enumeration of polynomial time truth-tables. We use the same complete set, but use the advice to either produce $\langle 0, x \rangle$ or $\langle 1, x \rangle$ on input x . There are again some cases to consider, but in this case we only need a subset of $\{0\}^*$ to diagonalize.

Yet reductions with advice and the induced completeness notions show familiar properties. Sets complete for EXP are complete under length-increasing, one-one reductions and are dense.

Theorem 10. *If A is complete for EXP under $\leq_m^{p/1}$ -reductions, then A is complete for EXP under length-increasing $\leq_m^{p/1}$ -reductions as well.*

Proof. Let $\{M_i\}_i$ be an enumeration of polynomial time many-one reductions that use one bit of advice. Let D be defined as follows. For $b \in \{0, 1\}$ if $|M^b(\langle b, i, x \rangle)| \leq |x|$ then let $\langle b, i, x \rangle \in D$ iff $x \notin A$. Else, let $\langle b, i, x \rangle \in D$ iff $x \in K$. Since A is complete for EXP, one of the reductions, say j must compute the reduction from D to A . For this reduction, it is never the case that $|M_j(\langle b, i, x \rangle)| \leq |x|$ when b is the correct advice for $|\langle b, i, x \rangle|$. Hence the reduction $f(x) = M_j(\langle b, i, x \rangle)$ where b is the correct advice for $|\langle b, i, x \rangle|$ is a $\leq_m^{p/1}$ length-increasing reduction from K to A .

Theorem 11. *Let A be complete for EXP under $\leq_m^{p/1}$ reductions, then A is dense.*

Proof. We construct a set W in EXP as follows. For $b \in \{0, 1\}$ and $x \in \{0, 1\}^*$. If there is an $x' < x$ such that $M_i^b(\langle b, i, x \rangle) = M_i^b(\langle b, i, x' \rangle)$, then $\langle b, i, x \rangle \in W \leftrightarrow \langle b, i, x' \rangle \notin W$. Otherwise, $\langle b, i, x \rangle \in W$. The set W is exponential time computable and dense (it has at least one of $\langle 0, i, x \rangle, \langle 1, i, x \rangle$ for every i and x). The reduction from W to A cannot have collisions for the correct advice, hence A must also be dense.

Theorem 12. *Let A be complete in EXP under $\leq_m^{p/1}$ -reductions, then A is also complete under $\leq_{m,1-1,li}^{p/1}$ -reductions.*

Proof. Let K be the canonical EXP-complete set. Let $K_0 = \{\langle 0^n, x \rangle \mid n \in \mathbb{N}, x \in K\}$. Clearly, K_0 is $\leq_{m,1-1,li}^p$ -complete in EXP. We show a $\leq_{m,1-1,li}^{p/1}$ -reduction from K_0 to A .

Let $\{M_i\}_i$ be an enumeration of polynomial time many-one reductions that take one bit advice. Let $M_i^b(x)$ denote the output of machine i on input x using advice b . First we use the constructions of Theorem 10 and Theorem 11 to find a reduction M_k from K_0 to A that is both length-increasing, and 1-1 on strings of the same length. That is, if b is the correct advice for $|x|$ then $(\forall x)[|M_k^b(x)| > |x|]$ and $(\forall x, y)[|x| = |y| \rightarrow M_k^b(x) \neq M_k^b(y)]$.

Without loss of generality, M_k runs in time n^k for almost all inputs, and hence for almost all x, y and b if $|y| > |x|^k$ then $M_k^b(x) \neq M_k^b(y)$. Now let $y_0(x) = \min\{y \mid (\exists \ell)[y = 0^{2^{(k+1)\ell} - |x|}]\}$ and define $f^b(x) = M_k^b(y_0(x))$.

First note that f is polynomial time computable. For every x the length of $y_0(x)$ is at most $2^k|x|$, so that the computation of M_k on $y_0(x)$ takes at most $2^{k^2}|x|^k$ time, which is polynomial in $|x|$. Next, if $x \neq x'$ then either $|y_0(x)| = |y_0(x')|$ in which case $f^b(x) \neq f^b(x')$, for b the correct advice for length $|y_0(x)|$ by properties of M_k , or $|y_0(x)| \neq |y_0(x')|$ but then, without loss of generality, assume $y_0(x')$ to be the longer string $|y_0(x')| \geq |y_0(x)|^k$, then $f^b(x) \neq f^{b'}(x')$ for b the correct advice for length $|y_0(x)|$ and b' the correct advice for length $|y_0(x')|$ by the fact that M_k is length increasing and n^k -time computable. In both cases f is 1-1. The advice given to f on input x is the advice belonging to M_k on input $y_0(x)$, which depends only on $|x|$.

However, unlike the uniform case, EXP has a $\leq_m^{p/1}$ -complete set that is P-bi-immune.

Theorem 13. *There exists a $\leq_m^{p/1}$ -complete set in EXP that is P-bi-immune.*

Proof. For immunity we construct a set A complete under $\leq_m^{p/1}$ -reductions. Let $\{M_i\}_i$ be an enumeration of polynomial time computable sets. If at some length n we find that one of the machines M_i with $i < n$ accepts a string $\{b, x\}$ for $b \in \{0, 1\}$ then we leave all strings $\{b, x\}$ out of A for this length and put $\{1-b, x\}$ in A iff $x \in K$. This construction is easily adapted to produce a bi-immune set by installing odd and even requirements and forcing a nonempty intersection with A as well as with \bar{A} by priority.

At the cost of more queries, in particular at the cost of adaptive queries, there exists a relation between completeness under reductions that use advice and uniform reductions as the following theorem shows.

Theorem 14. *Every set A that is $\leq_m^{p/1}$ complete in EXP is also \leq_T^p -complete in EXP.*

Proof. (sketch) For this proof we use the representation of exponential time machines with tableaux. Let A be some $\leq_m^{p/1}$ complete set. Suppose that we want to compute $x \in B$ for B some complete set. Let $B = L(M^{A/1})$. Consider an exponential size rectangle that is an encoding of the computation $x \in B$. We call this rectangle the *tableau* of the computation $x \in B$. Say, a *bit* in this computation has position i, j , where $1 \leq i, j \leq 2^{p(n)}$, and assume that the *final* bit in this computation is 0 or 1 representing reject and accept respectively. Every bit in the tableau can be computed using $M^{A/1}$. By padding i, j , where $1 \leq i, j \leq 2^{p(n)}$, we can ensure that input $\langle x, i, j \rangle$ is of the same length for every $\langle x, i, j \rangle$. Then $M^{A/1}$, given the right advice bit, computes every bit of the tableau of $x \in B$ correctly. We will call the tableau representing the result of the computation using advice 0 the 0-tableau and the tableau representing the result of the computation using advice 1 the 1-tableau. It follows that if the final bits in the 0-tableau and the 1-tableau are the same, then we know the answer to $x \in B$.

If the final bits are not the same, then there must be an inconsistency for exactly one of the two tableaux (the one computed with the incorrect advice). Given the correct advice bit we can in exponential time recover the first inconsistency in the tableau computed using the reduction to A with the incorrect advice. This is done as follows: we suppose that 0 is the correct advice, and using this advice, we retrieve the first sequence of bits that is inconsistent when the 1-tableau for $x \in B$ is computed. Then, using advice 1, we recompute these bits. If 0 is indeed the correct advice, then, since the computation using 1 as advice is exponential time, we must indeed find an inconsistent sequence of bits. On the other hand, if 1 is the correct advice, such a sequence of bits *does not exist* and using 1 as the advice, we find a consistent sequence of bits on the indicated position (as we would find on any position).

With a slightly more complicated proof the above theorem can be extended to reductions using up to $c \log n$ advice. The different advice strings then give rise to a polynomial number of (possibly disagreeing) tableaux. With help of the one correct advice, an exponential time computation, and thus a polynomial time reduction, can point out the first errors in the other computations, thus providing proof for the correct result.

The tableau technique is a nonrelativizing technique. It is no surprise that this theorem does not relativize. We make this explicit in the following theorem.

Theorem 15. *There exists an oracle A and an set B that is $\leq_m^{p/1, A}$ -complete for EXP^A but not $\leq_T^{p, A}$ -complete for EXP^A .*

Proof. (sketch) For this proof we use the construction of an oracle appearing in [BvMFT00]. Here, a set A is constructed such that EXP^A has a Turing complete sets that is not autoreducible. In particular there is a subset S of $\{0^n\}$ such that $A - S$ is no longer Turing complete. The Turing reduction that makes A complete is a very simple one. If $0^{|x|} \in A$ then $x \in K \leftrightarrow \langle 0, x \rangle \in A$ else $x \in K \leftrightarrow \langle 1, x \rangle \in A$. Of course this is a $\leq_m^{p/1}$ reduction and $A - S$ remains $\leq_m^{p/1}$ complete under this reduction

The fact that many-one completeness with one bit of advice can be simulated using more queries, extends to the delta levels of the polynomial hierarchy. Here the reduction does not even have to be adaptive as the following theorem shows.

Theorem 16. *For every k and every set A in Δ_k^p , if A is $\leq_m^{p/1}$ -complete, then A is \leq_{tt}^p -complete.*

Proof. We give the proof here for P^{NP} . The proof for higher levels is analogous. Let $L_1 = \{\langle \phi, i \rangle \mid \text{the } i\text{-th bit in the lexicographically least assignment that makes } \phi \text{ true is } 1\}$. Then clearly, since ODDMAXBIT is complete for P^{NP} , so is L_1 . Let A be a $\leq_m^{p/1}$ complete set for P^{NP} . We show how to compute $\phi \in \text{ODDMAXBIT}$ using a truth table reduction to A .

Suppose then that M computes the reduction from L_1 to A given advice b , and let $M^b(\langle \phi, i \rangle)$ be the output of the reduction, i.e. for the correct advice b it holds that $M^b(\langle \phi, i \rangle) \in A$ if and only if $\langle \phi, i \rangle \in L_1$. Without loss of generality, we assume that ϕ has n variables and use a padded version of L_1 , such that $\langle \phi, 0 \rangle, \dots, \langle \phi, n \rangle$ are all of the same length, i.e. use the same advice. Now compute

$$\begin{aligned}\omega_0 &= \langle M^0 \langle \phi, 0 \rangle \rangle, \dots, \langle M^0 \langle \phi, n \rangle \rangle \\ \omega_1 &= \langle M^1 \langle \phi, 0 \rangle \rangle, \dots, \langle M^1 \langle \phi, n \rangle \rangle\end{aligned}$$

and check that either ω_0 or ω_1 makes ϕ true. At least one of these must be true as 0 or 1 must be the correct advice and return the bits of the lexicographically least assignment. This implies that the other advice either returns an assignment that does not make ϕ true, or an assignment that is not lexicographically smaller. In any case, we now have the lexicographically least assignment in hand.

This has some interesting consequences. We don't know whether theorem 14 can be strengthened to truth-table reductions. However there are two cases.

1. If $\leq_m^{p/1}$ -completeness implies \leq_{tt}^p -completeness on EXP then $\text{EXP} \neq \text{EXPSPACE}$.
2. If $\leq_m^{p/1}$ -completeness does *not* imply \leq_{tt}^p -completeness on EXP , then $\text{PH} \neq \text{EXP}$.

Acknowledgments

We thank John Hitchcock, Eric Allender, and Stephen Fenner for useful discussions and feedback. We thank John Hitchcock for his observation that the hypothesis in our length increasing result for NP can be weakened to bi-immunity instead of measure zero.

References

- [AB93] M. Agrawal and S. Biswas. Polynomial isomorphism of 1-L complete sets. In *Proc. Structure in Complexity Theory 7th Annual Conference*, pages 75–80, San Diego, California, 1993. IEEE Computer Society Press.
- [ABK⁺02] Eric Allender, Harry Buhrman, Michal Koucký, Dieter van Melkebeek, and Detlef Ronneburger. Power from random strings. In *FOCS*, pages 669–678. IEEE Computer Society, 2002.
- [Agr02] Manindra Agrawal. Pseudo-random generators and structure of complete degrees. In *IEEE Conference on Computational Complexity*, pages 139–147, 2002.
- [All88] E. Allender. Isomorphisms and 1-L reductions. *J. Computer and System Sciences*, 36(6):336–350, 1988.
- [AS84] K. Ambos-Spies. p-mitotic sets. In E. Börger, G. Hasenjäger, and D. Roding, editors, *Logic and Machines, Lecture Notes in Computer Science 177*, pages 1–23. Springer-Verlag, 1984.
- [BDG88] J. Balcázar, J. Díaz, and J. Gabarró. *Structural Complexity I*. Springer-Verlag, 1988.
- [BH77] L. Berman and H. Hartmanis. On isomorphisms and density of NP and other complete sets. *SIAM J. Comput.*, 6:305–322, 1977.
- [BHT91] H. Buhrman, S. Homer, and L. Torenvliet. On complete sets for nondeterministic classes. *Math. Systems Theory*, 24:179–200, 1991.
- [BM95] H. Buhrman and E. Mayordomo. An excursion to the kolmogorov random strings. In *Proceedings Structure in Complexity Theory, 10th annual conference (STRUCTURES95)*, pages 197 – 205, Minneapolis, 1995. IEEE Computer Society Press.
- [BST93a] H. Buhrman, E. Spaan, and L. Torenvliet. Bounded reductions. In K. Ambos-Spies, S. Homer, and U. Schöning, editors, *Complexity Theory*, pages 83–99. Cambridge University Press, December 1993.
- [BST93b] H. Buhrman, E. Spaan, and L. Torenvliet. The relative power of logspace and polynomial time reductions. *Computational Complexity*, 3(3):231–244, December 1993.
- [BT99] H. Buhrman and L. Torenvliet. Complicated complementations. In *Proceedings 14th IEE Conference on Computational Complexity*, pages 227–236. IEEE Computer Society Press, May 1999.
- [BT04] H. Buhrman and L. Torenvliet. Separating complexity classes using structural properties. In *Proceedings 19th IEE Conference on Computational Complexity*, pages 130–138. IEEE Computer Society Press, 2004.
- [BT05] Harry Buhrman and Leen Torenvliet. A post’s program for complexity theory. In *Bulletin of the EATCS 85*, pages 41–51. 2005.
- [BvMFT00] H. Buhrman, D. van Melkebeek, L. Fortnow, and L. Torenvliet. Using autoreducibility to separate complexity classes. *Siam Journal on Computing*, 29(5):1497–1520, 2000.
- [FFK92] S. Fenner, L. Fortnow, and S.A. Kurtz. The isomorphism conjecture holds relative to an oracle. In *Proc. 33rd IEEE Symposium Foundations of Computer Science*, pages 30–39, 1992.
- [GH88] K. Ganesan and S. Homer. Complete problems and strong polynomial reducibilities. In *Proc. Symposium on Theoretical Aspects of Computer Science, Springer Lecture Notes in Computer Science 349*, pages 240–250. Springer-Verlag, 1988.

- [HH91] J. Hartmanis and L. Hemachandra. One-way functions and the non-isomorphism of NP-complete sets. *Theoretical Computer Science*, 81(1):155–163, 1991.
- [HKR93] S. Homer, S. Kurtz, and J. Royer. A note on many-one and 1-truth table complete sets. *Theoretical Computer Science*, 115(2):383–389, July 1993.
- [HP04] J.M. Hitchcock and A. Pavan. Hardness hypotheses, derandomization, and circuit complexity. In *24th Conference on Foundations of Software Technology and Theoretical Computer Science*, pages 336–347. Springer Verlag, December 2004.
- [HP06] John M. Hitchcock and Aduri Pavan. Comparing reductions to np-complete sets. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, *ICALP (1)*, volume 4051 of *Lecture Notes in Computer Science*, pages 465–476. Springer, 2006.
- [HS92] S. Homer and A.L. Selman. Oracles for structural properties: the isomorphism problem and public-key cryptography. *Journal of Computer and System Sciences*, 44(2):287–301, 1992.
- [HS01] Steven Homer and Alan L. Selman. *Computability and Complexity Theory*. Springer-Verlag, New York, 2001.
- [KL80] R. Karp and R. Lipton. Some connections between nonuniform and uniform complexity classes. In *Proc. 12th ACM Symposium on Theory of Computing*, pages 302–309, 1980.
- [KMR89] S. Kurtz, S. Mahaney, and J. Royer. The isomorphism conjecture fails relative to a random oracle. In *Proc. 21st Annual ACM Symposium on Theory of Computing*, pages 157–166, 1989.
- [Kre88] M. Krentel. The complexity of optimization problem. *J. Computer and System Sciences*, 36:490–509, 1988.
- [LLS75] R. Ladner, N. Lynch, and A. Selman. A comparison of polynomial time reducibilities. *Theoretical Computer Science*, 1:103–123, 1975.
- [LM94] J. Lutz and E. Mayordomo. Cook versus Karp-Levin: Separating reducibilities if NP is not small. In *STACS 1994, Lecture Notes in Computer Science*. Springer-Verlag, 1994.
- [Pap94] C.H. Papadimitriou. *Computational Complexity*. Addison Wesley, 1994.
- [Ron04] Detlef Ronneburger. *Kolmogorov Complexity and Derandomization*. PhD thesis, Rutgers University, New Brunswick, NJ, October 2004.
- [Wat87] O. Watanabe. A comparison of polynomial time completeness notions. *Theoretical Computer Science*, 54:249–265, 1987.
- [You90] P. Young. Juris Hartmanis: Fundamental contributions to the isomorphism problems. In A.L. Selman, editor, *Complexity Theory Retrospective*, pages 108–146. Springer-Verlag, 1990.

A Some Proofs

A.1 Proof of Theorem 5

We construct sets A , C , and D , such that $C = K_{\text{NEXP}}^A - D$ is complete for NEXP^A , but not via length-increasing reductions. The construction has two phases, a diagonalization phase in which we take care that $\{0\}$ is not reduced to C via a many-one reduction that is length increasing on almost all strings, and an encoding phase, in which the strings in D that are also in K_{NEXP}^A are encoded into the oracle in such a way that a nondeterministic exponential time set, here C , can compute them on inputs that are exponentially smaller. We use a fast growing but easy to compute function $b(i)$ which is at least double exponential, i.e., $b(i+1) > 2^{2^{b(i)}}$.

Phase 1: Diagonalization - In this phase we construct D , and part of A . Additional notation: For integers i and j , and sets X and Y , let $q_{\text{next}}(i, j, X, Y)$ denote the first query of the form $q = \langle 0, j', x \rangle$, computed by $f^X(0^{b(i)})$ with $|q| > b(i)$, $x \notin Y$, $q \notin X$, and $j' \leq j$, if any, and λ otherwise.

- 1: **Stage 0:** $A = \emptyset$, $D = \emptyset$;
- 2: **Stage i :** set $j = 1^{i \log b(i)}$; $D_i = \emptyset$; $A_i = \emptyset$;
- 3: **while** ($q = q_{\text{next}}(i, j, A \cup A_i, D_i) \neq \lambda$) **do**
- 4: $A_i = A_i \cup \{\langle 0, c, \pi_3(q) \rangle \mid 0^{i \log b(i)} \leq c \leq j\}$;
- 5: $D_i = D_i \cup \{\pi_3(q)\}$; $j = j - 1$;
- 6: **end while**
- 7: **if** $|f_i^{A \cup A_i}(0^{b(i)})| > b(i)$ **then**
- 8: $D = D \cup D_i \cup \{f_i(0^{b(i)})\}$;
- 9: $A = A \cup A_i \cup \{\langle 0, c, f_i(0^{b(i)}) \rangle \mid 0^{i \log b(i)} \leq c \leq j\}$;
- 10: **end if**

Phase 2: Coding - here we make adjustments to the oracle A such that $K^A - D$ remains incomplete via length increasing reductions, however strings in D are reencoded in A such that a NEXP machine can recover these strings on inputs of logarithmic lengths.

- 1: **Stage x :**
- 2: **if** $x \in K^A \cap D$ **then**
- 3: Let $c = \max\{i \mid \langle 0, i, x \rangle \in A\}$;
- 4: Let $k = b(i)^i$ where $b(i) \leq |x| \leq b(i)^i$;
- 5: Let $y = \min\{z \mid |z| = k^2 \wedge \langle 1, c, z \rangle \notin Q(f_i^A(0^{b(i)})) \cup \{Q(N_K^A(v)) \mid v \leq x\}\}$
- 6: $A = A \cup \{\langle 1, c, y \rangle\}$;
- 7: **end if**

Interleaving The diagonalization phase and the coding phase can be executed simultaneously, but interleaved. The diagonalization at stage i assumes that the oracle is fixed below length $b(i)$. Therefore the coding of all such strings has to be done before this stage starts. This means that the coding of all x 's in $K^A \cap D$ that require changes to the oracle at lesser length have to be encoded before stage i . The x 's in $\cup\{D_j \mid j < i\}$ are of length at most $b(i-i)^{i-1}$. To determine whether these x 's are also in K^A , queries must be made to A at

lengths at most $2^{b(i-1)^{i-1}}$ (note that this may be changed by the encoding itself, but not by stages $> i$ of the diagonalization, moreover this encoding stabilizes by the fact that queries of previously encoded strings are avoided). Then these x 's are encoded at length $|x|^2$ which is at maximum $(b(i-1)^{i-1})^2$. However $(\forall^\infty i)[b(i-1)^{(i-1)^2} < 2^{b(i-1)^{i-1}} < b(i)]$, since $b(i)$ is double exponential. It follows that diagonalization and coding can be safely interleaved by first doing an entire diagonalization stage, and then coding the strings that entered D during that stage.

Lemma 1 (Room to Diagonalize). *Whenever strings need to be added to A during the diagonalization phase, there are enough strings not yet determined.*

Proof. The diagonalization phase during stage i adds strings of length between $b(i)$ and $b(i)^i$. The number of these strings is bounded by $b(i)^i \times \sum_{j=0}^{b(i)^i} 0^{b(i)^i} j$, which is less than $b(i)^{3i}$. For all but finitely many x this is less than $2^{b(i)^i}$, the number of strings not yet set in the oracle at this length.

Lemma 2 (Room to encode). *Whenever strings need to be added to A during the encoding phase, there are enough strings not yet determined.*

Proof. Whenever a string x needs to be encoded, we search for a string y of length $|x|^2$. Such that $\langle 1, c, x \rangle$ is not yet in the oracle. Note that strings added by the diagonalization phase are all of the form $\langle 0, u, v \rangle$, so these do not matter. Other strings that could be prohibited from entering the oracle are strings queried in a computation of $M_{K_{\text{NEXP}}}^A$, on inputs $y < x$. There are less than $2^{|x|+1}$ of these strings, each giving rise to at most $2^{|y|}$ queries. It follows that this number is bounded by $2^{2|x|+1}$ which is much less than the $2^{|x|^2}$ strings available for almost all x .

The complete set C is defined as follows.

- 1: **input** $\langle i, x \rangle : i \in \{0, 1\}$;
- 2: **if** $i = 0$ and $x \in K^A - D$ **then** accept
- 3: **end if**
- 4: **if** $(\exists y)[|y| = 2^{2|x|} \wedge \langle 1, x, y \rangle \in A]$ **then** accept
- 5: **end if**
- 6: reject

Lemma 3. $C \in \text{NEXP}^A$.

Proof. It follows more or less straightforward from the construction and the way diagonalizing and coding is interleaved that computing whether an input is in D can be done in deterministic exponential time. Observe that strings for which nondeterministic exponential time computations must be simulated are all exponentially smaller. $x \in K^A$ can be decided in nondeterministic linear exponential time, and an oracle query of length $2^{2|x|}$ can be built in nondeterministic linear exponential time as well.

Lemma 4. 0^* does not reduce to C via a reduction that is almost always length increasing.

Proof. Suppose it does, then this length increasing reduction has some index i where $b(i+1)$ is large enough to satisfy all room to diagonalize/code conditions and moreover $M_i(0^{b(i)}) > b(i)$. At stage i the oracle is fixed so that the output $M_i(0^{b(i)})$ is in D and hence not in C , a contradiction.

The final part is the definition of the reduction F from K^A to C .

- 1: **input** x
- 2: Let $k = \max\{i \mid b(i) \leq |x|\}$
- 3: **if** $\langle 0, 0^{i \log b(i)}, x \rangle \notin A$ **then** output $\langle 0, x \rangle$;
- 4: **else**
- 5: Let $\ell = \max\{j \mid \langle 0, j, x \rangle \in A\}$;
- 6: output $\langle 1, \ell \rangle$;
- 7: **end if**

The final lemma for this construction then says that C remains complete.

Lemma 5. $K^A \leq_m^p C$.

Proof. If $x \notin D$ then $x \in K^A$ iff $x \in C$. On the other hand in this case no string of the form $\langle 0, y, x \rangle$ is in A by construction, so the reduction, which is identity in this case works. If $x \in D \cap K^A$ then for the maximum c such that $\langle 0, c, x \rangle$ is in A there exists a string y of length $|x|^2$ with $\langle 1, y, c \rangle \in A$, whence $c \in C$. If $x \in D - K^A$ then no such y is in A , whence $c \notin C$.

A.2 Proof of Theorem 8

We use a construction appearing in [BT99] in which the oracle is created from an infinite Kolmogorov random string. The language that will have the property stated in the theorem will be the following. For all oracles X define $D_X = \{x \mid (\exists y)[|y| = n^{2c} \wedge xy \in X]\}$. Let Y be an infinite string that has the property that $(\forall n)[K(Y_{[1..n]}) > n^c]$. We will construct A by stages. A_{s+1} is the oracle defined by the end of stage s and $A_{s+1} \subseteq A_s$ for all s . The oracle A may have 2^n strings of length $n + n^{2c}$. We now describe the initial oracle A_0 .

To encode a string x into D , we need a sequence of $|x|^{2c}$ bits. For this encoding we use substrings of Y . Up to $|x|$ we have used bits of Y for $\sum_{i < |x|} 2^i \leq 2^{|x|}$ strings, so if we let the substrings of Y that encode strings of length $|x|$ start at the $2^{|x|} \times |x|^{2c}$ th bit of Y , there will never be a conflict (two strings in A taking their encoding from the same substring of Y , note that strings smaller than x require less bits for encoding). Moreover, we have sequence from the $2^{|x|} \times |x|^{2c}$ th bit up until the $2^{|x|+1} \times |x|^{2c}$ th bit of $2^{|x|}$ substrings of length $|x|^{2c}$ that can be used to encode strings of length $|x|$. The encoding of strings of length $|x|+1$ then starts at the $2^{|x|+1} \times |x+1|^{2c}$ th bit of Y , which is quite a bit further along. For A_0 up to the i th string of length m we use a substring of Y , say $Y_{A_0}^{m,i}$ of length $2^m m^{2c} + i m^{2c}$ bits. The incompressibility of Y implies that for some constant d and all m and i it holds that $K(Y_{A_0}^{m,i}) \geq 2^m m^{2c} + i m^{2c} - d$. This is the property we will use to prove correctness of our construction. This ends the description of the initial oracle A_0 . We shall next describe a stage construction of the oracle that has the desired properties.

At each stage s we will decide the membership of the string x_s in D^A , where x_s is the s th string in the lexicographical ordering of Σ^* . Deciding membership of x_s is deciding whether to remove $x_s y$ from A where $x_s y$ is the only string that is an extension of x_s currently in A . The Kolmogorov property that makes the entire construction work is that a (nondeterministic) machine that *rejects* some string x_s with oracle A_s *must also reject* x_s with oracle A or else it will allow us—via a description of the position of a query on the least accepting path—to describe some initial segment of Y using significantly less bits than the length of this segment, which then violates the aforementioned Kolmogorov property.

Let $\{M_i\}_i$ be an enumeration of all nondeterministic Turing machines, where 2^{n^c} is the time bound on inputs of length n for all machines in the enumeration. The construction maintains two sets of requirements. First, a set U_s of yet unsatisfied requirements to which occasionally a new element is added and from which satisfied requirements are removed. Second, a set V_s in which satisfied requirements are kept. Sometimes a satisfied requirement will be moved from V_s to U_s at which time it will become unsatisfied again. Every requirement corresponds to a language in $\text{NTIME}(2^{n^c}) \cap \text{co-NTIME}(2^{n^c})$, represented by nondeterministic 2^{n^c} -time bounded machines M_i and M_j . Even requirements $R_{2\langle i,j \rangle}$ will represent the need to establish a nonempty intersection of $L(M_i) \cap L(M_j)$ with D , whereas odd requirements $R_{2\langle i,j \rangle+1}$ will represent the need to establish a nonempty intersection of $L(M_i) \cap \overline{L(M_j)}$ with \overline{D} .

We will prove that the construction of A meets the following requirements for all $\langle i, j \rangle$.

1. $R_{2\langle i,j \rangle}$: $\|L(M_i^A)\| = \infty \implies [[L(M_i^A) \neq \overline{L(M_j^A)}] \vee [L(M_i^A) \cap D_A \neq \emptyset]]$.
2. $R_{2\langle i,j \rangle+1}$: $\|L(M_i^A)\| = \infty \implies [[L(M_i^A) \neq \overline{L(M_j^A)}] \vee [L(M_i^A) \cap \overline{D_A} \neq \emptyset]]$.

There are two ways to fulfill these requirements. Either maintain a difference between $L(M_i)$ and $\overline{L(M_j)}$, i.e., the pair M_i, M_j does not represent a language in $\text{NTIME}(2^{n^c}) \cap \text{co-NTIME}(2^{n^c})$, or maintain a string in the intersection of $L(M_i)$ and D , respectively \overline{D} .

A requirement R_e with $e = 2\langle i, j \rangle$, or $e = 2\langle i, j \rangle + 1$ is *active* at stage s if $x_s \in L(M_i^{A_s})$. Now the construction can be described as follows.

- 1: **stage** s :
- 2: **if** no requirement is active **then**
- 3: $A_{s+1} = A_s$;
- 4: $V_{s+1} = V_s$; $U_{s+1} = U_s$;
- 5: **else**
- 6: let e be the least active requirement in U_s ;
- 7: **if** e is even **then**
- 8: $U_{s+1} = U_s - \{e\}$;
- 9: $V_{s+1} = V_s \cup \langle e, x_s \rangle$; $A_{s+1} = A_s$;
- 10: **else**
- 11: **if** $\{\langle i, x \rangle \in V_s \mid (i < e) \wedge (2^{|x|^c} > |x_s|)\} = \emptyset$ **then**
- 12: $V_{s+1} = V_s - \{\langle i, x \rangle \in V_s \mid i > e\}$
- 13: $U_{s+1} = U_s \cup \{i \mid (i > e) \wedge (\exists x)[\langle i, x \rangle \in V_s]\} - \{e\}$;

```

14:      $A_{s+1} = A_s - \{ \langle x_s, y_s \rangle \};$ 
15:   else
16:      $A_{s+1} = A_s; V_{s+1} = V_s; U_{s+1} = U_s;$ 
17:   end if
18: end if
19: end if
20: if  $s = 2^k$  then  $U_{s+1} = U_{s+1} \cup \{k\}.$ 
21: end if

```

We will now prove correctness of our construction in a series of lemmas. The key lemma in the proof is the following.

Lemma 6. $(\forall^\infty s)(\forall e < \log s)[x_s \notin L(M_e^{A_s}) \implies x_s \notin L(M_e^A)]$

Proof. Suppose not. Let e and s be such that $e < \log s$ and $x_s \in L(M_e^A) - L(M_e^{A_s})$. Let s' be minimal such that $s' \geq s$ and $x_s \in L(M_e^{A_{s'+1}}) - L(M_e^{A_{s'}})$. By construction $A_{s'} - A_{s'+1} = \{x_{s'}y_{s'}\}$. The string $x_{s'}y_{s'}$ must be queried in *any* accepting computation of $M_e^{A_{s'+1}}$ on input x_s . Otherwise $M_e^{A_{s'}}$ would also have an accepting computation on input x_s contradicting the assumption. On input x_s , machine M_e can only query strings of length less than or equal to $2^{\lfloor |x_s|^c \rfloor}$. Moreover $\|\{y \mid y \in A_0 - A_{s'+1}\}\| < \log s'$ since for every such y there is an odd index in $U_t - (U_{t+1} \cup V_{t+1})$ for some $t < s'$ and there are no more than $\log s'$ indices in $\cup\{U_t \mid t \leq s'\}$. Let n be such that $n^{2c} > |x_{s'}|^{2c}$. We will show how to construct the first $2^{n+1}n^{2c}$ bits of Y using significantly less bits and hence arrive at a contradiction. Suppose that we have $2^{n+1}n^{2c} - |x_{s'}|^{2c}$ bits of Y that describe the initial segment of Y used to encode A_0 for all substrings of Y up to and including strings of length $n + n^{2c}$, *except* $y_{s'}$ (For strings of length $n + n^{2c}$ there is a corresponding substring of length n^{2c} in Y as explained above. So $y_{s'}$ in Y , $x_{s'}y_{s'}$ in A_0). Furthermore suppose that we have a list of at most $\log s'$ strings x_{i_1}, \dots, x_{i_k} that says which $x_{i_j}y_{i_j}$ are in $A_0 - A_{s'}$. Note that no strings greater than $x_{s'}y_{s'}$ are in $A_0 - A_{s'}$. Finally let $q < 2^{\lfloor |x_s|^c \rfloor}$ be the index of the query $x_{s'}y_{s'}$ in the leftmost accepting computation of $M_e^{A_{s'+1}}(x_s)$. Note that q requires at most $|x|^c$ bits. Now we can construct the first $2^{n+1}n^{2c}$ bits of Y from $2^{n+1}n^{2c} - |x_{s'}|^{2c} + \log s' \times |x_{s'}| + |x_{s'}'| + |x_s|^c + O(1)$ bits of information. We arrive at a contradiction for all but finitely many s . The difference between $|x_s|^{2c}$ and $\log s' \times |x_{s'}| + |x_{s'}'| + |x_s|^c + O(1)$ will outgrow any constant, and so the complexity assumption on Y will be violated.

Lemma 7. $(\forall e)[\|\{s \mid (\exists x)[\langle e, x \rangle \in V_{s+1} - V_s]\}\| < \infty]$

Proof. Whenever $\langle e, x \rangle$ is in $V_{s+1} - V_s$ there is a smaller index that is moved from U_{s+1} either to oblivion or to V_{s+1} . This means that 0 can enter V_s for only one s and there is no t such that $\langle 0, x \rangle$ is in $V_t - V_{t+1}$. By induction for $i < e$, let $\#i$ be the number of times that index i is in $V_{t+1} - V_t$ for some t and x . Then it is clear that $\#e \leq \sum_{i=0}^{e-1} (1 + \#i)$ which is finite for all e .

Corollary 2. $(\forall e)[(\exists^\infty s)[e \in U_s] \implies (\forall^\infty s)[e \in U_s]]$

Lemma 8. *If $\|L(M_e^A)\| = \infty$ then $L(M_e^A) \cap D_A \neq \emptyset$*

Proof. Let e be such that $\|L(M_e^A)\| = \infty$ and $L(M_e^A) \cap D_A = \emptyset$. If $2e$ in V_{s+1} then $L(M_e^{A_{s+1}}) \cap D_{A_{s+1}} \neq \emptyset$. Since $L(M_e^A) \cap D_A = \emptyset$ there are infinitely many s such that $2e \notin V_s$. It follows from Corollary 2 that $2e \in U_s$ for almost all s . Consequently there are infinitely many s such that $x_s \in L(M_e^A)$ where $2e \in U_s$ and $e < \log s$. At stage s there are two possibilities.

1. $x_s \in L(M_e^{A_s})$
2. $x_s \notin L(M_e^{A_s})$.

In case 1, since $x_s \notin D_A$ there is some $e' < e$ such that $2e' + 1 \in U_{s+1} - U_s$. Hence case 1 can appear only finitely often. Case 2 must appear infinitely often. Consider a stage where case 2 appears. Then we have $e < \log s$ and $x_s \in L(M_e^A) - L(M_e^{A_s})$, so it follows from Lemma 6 that this can also appear only finitely often, a contradiction.

Lemma 9. *If $L(M_i^A) = \overline{L(M_j^A)}$ and $\|L(M_i^A)\| = \infty$ then $L(M_i^A) \cap \overline{D_A} \neq \emptyset$.*

Proof. We will first argue that for almost all i and j , if $2\langle i, j \rangle + 1$ is in $U_s - (U_{s+1} \cup V_{s+1})$ then $L(M_i^A) \cap \overline{D_A} \neq \emptyset$. Note that a particular language in $\text{NTIME}(2^{n^c}) \cap \text{co-NTIME}(2^{n^c})$ is represented by infinitely many pairs $\langle i, j \rangle$. Suppose that for some (large) s the index $2\langle i, j \rangle + 1$ is in $U_s - (U_{s+1} \cup V_{s+1})$. This means that $x_s \in \overline{D_A}$ and $x_s \notin L(M_j^{A_s})$. It follows from Lemma 6 that $x_s \notin L(M_j^A)$ and hence from the assumption $L(M_i^A) = \overline{L(M_j^A)}$ that $x_s \in L(M_i^A)$. If $(\forall^\infty s)(\exists x)[\langle e, x \rangle \in V_s]$ then let t be maximal so that $e \notin V_t$. Since $A_t = A_{t+1}$, it is then the case that $x_t \in L(M_i^{A_{t+1}}) \cap L(M_j^{A_{t+1}})$ and moreover since $\langle e, x_t \rangle \in V_u$ for all $u \geq t+1$, subsequent changes to the oracle are made only at lengths greater than $2^{|x_t|^c}$ whence $x_t \in L(M_i^A) \cap L(M_j^A)$. In this case there is nothing to prove. Finally, let $e = 2\langle i, j \rangle + 1$ be such that $L(M_i^A) = \overline{L(M_j^A)}$ and $\|L(M_i^A)\| = \infty$ and $L(M_i^A) \subseteq D_A$ and moreover $(\forall^\infty s)[e \in U_s]$. By Corollary 2 also $(\forall^\infty s)(\forall e' \leq e)[e' \in U_s \implies e' \in U_{s+1}]$, i.e., no requirement of higher priority will be satisfied at stage s . Consider only two stages where $x_s \in L(M_i^A)$ and $(\forall e' \leq e)[e' \in U_s \implies e' \in U_{s+1}]$. At any of these stages there are two possibilities.

1. $x_s \in L(M_i^{A_s})$
2. $x_s \notin L(M_i^{A_s})$.

In the first case $2\langle i, j \rangle + 1 \notin U_{s+1}$ contradicting the assumption and in the second case $x_s \in L(M_i^A) - L(M_i^{A_s})$ so this case can occur only finitely often according to Lemma 6

The following lemma concludes the proof.

Lemma 10. $\|D_A\| = \infty$ and $\|\overline{D_A}\| = \infty$.

Proof. This is immediate from the fact that both sets have nonempty intersections with languages in $\text{NTIME}(2^{n^c}) \cap \text{co-NTIME}(2^{n^c})$ at infinitely many different points x_s .

A.3 Detailed proof of Theorem 14

Proof. Let A be some $\leq_m^{p/1}$ complete set, let K be the standard many-one complete set. We will create a Turing reduction from K to A by using a few intermediate sets in EXP

These sets will be based on the tableau representation of the exponential time machine computing $x \in K$. Let the EXP machine computing K be M_K . We assume the running time of M_K is $2^{p(n)}$ for some polynomial p . Consider an exponential size rectangle that is an encoding of the computation $x \in K$. We call this rectangle the *tableau* of the computation $x \in K$. Say, a *bit* in this computation has position i, j , where $1 \leq i, j \leq 2^{p(n)}$, and assume that the *final* bit in this computation is 0 or 1 representing reject and accept respectively. Moreover, by padding i, j , where $1 \leq i, j \leq 2^{p(n)}$, we can ensure that for every x , $\langle x, i, j \rangle$ is of the same length for every $\langle i, j \rangle$. (Namely, i, j will be binary strings between $0^{p(n)}1$ and $10^{p(n)}$.)

Now consider the language B , the set of $\langle i, j, x \rangle$ with the following conditions:

1. $\langle i, j \rangle$ are each padded to length $p(|x|) + 1$
2. $\langle i, j \rangle$ bit in the tableau for $M_K(x)$ is 1

First notice that $B \in \text{EXP}$, and as A is $\leq_m^{p/1}$ -complete, there is a polynomial time machine, M_B^b , where for correct the advice b , $\langle i, j, x \rangle \in B \leftrightarrow M_B^b(\langle i, j, x \rangle) \in A$. Also notice there is a many-one reduction from K to B , i.e. $x \in K \leftrightarrow \langle 10^{p(|x|)}, 10^{p(|x|)}, x \rangle \in B$. Now for all i, j between $0^{p(n)}1$ and $10^{p(n)}$, $\langle i, j, x \rangle \in B$ corresponds to the $\langle i, j \rangle$ bit of $M_K(x)$ and $|\langle i, j, x \rangle| = |\langle i', j', x \rangle|$. This means that the same advice bit is used by M_B^b on all $\langle i, j, x \rangle$ to compute the bits of the exponential computation by M_K . Since we would like to remove the advice consider the computation of M_B^0 and M_B^1 over all $\langle i, j, x \rangle$ with i, j between $0^{p(n)}1$ and $10^{p(n)}$. Let the bits computed by M_B^0 be the 0-tableau and the bits by M_B^1 be the 1-tableau. Note that if the final bits in the 0-tableau and the 1 tableau are the same, then we know the answer to $x \in K$.

If the final bits are not the same, then there must be an inconsistency for exactly one of the two tableaux (the one computed with the incorrect advice). Now we will show how to recover the correct advice using the inconsistency in the tableau. Consider the set B' defined as:

$$B' = \{ \langle b, i, j, x \rangle \mid [\langle i, j, x \rangle \in B \wedge M_B^b(\langle i, j, x \rangle) \in A] \vee [\langle i, j, x \rangle \notin B \wedge M_B^b(\langle i, j, x \rangle) \notin A] \}$$

For the correct advice, b , to the reduction M_B , $\langle b, i, j, x \rangle \in B'$ for all i, j between $0^{p(n)}1$ and $10^{p(n)}$. For the incorrect advice, b' , there is some $\langle i, j \rangle$ with $\langle b', i, j, x \rangle \notin B'$.

We will also need to find this inconsistency in B' . Consider the set \hat{B} . Here we add in all $\langle b, i', j', x \rangle$, where $\langle i', j' \rangle$ is less than $\langle i, j \rangle$, where $\langle i, j \rangle$ is the location of the first inconsistency with M_K .

$$\hat{B} = \{ \langle b, i', j', x \rangle \mid \langle i', j' \rangle \text{ is less than the minimal } \langle i, j \rangle \text{ with } \langle b, i, j, x \rangle \notin B' \}$$

Notice that both B' and \hat{B} are in EXP, therefore there are polynomial time reductions from B' and \hat{B} to A which use one bit of advice. Let $M_{B'}^b$ and $M_{\hat{B}}^b$ be these reductions respectively.

We are now ready to define the polynomial time Turing machine with oracle access to A which computes $x \in K$.

On input x

1. Compute $p(|x|)$ where $2^{p(|x|)}$ is the running time of M_K . (Recall that M_K is the exponential machine computing K .)
2. Compute $q_0 = M_B^0(10^{p(|x|)}, 10^{p(|x|)}, x)$ and $q_1 = M_B^1(10^{p(|x|)}, 10^{p(|x|)}, x)$, where M_B^b is the polynomial time reduction from B to A using advice b .
3. Ask $q_0 \in A$ and $q_1 \in A$
 - If both yes ACCEPT
 - Else if both no, REJECT
 - Else continue
4. Now we will assume that 0 was the correct advice and that $M_B^0(10^{p(|x|)}, 10^{p(|x|)}, x) \in A$ is the right answer. This assumption also implies that there is some $\langle i, j \rangle$ element $\langle 1, i, j, x \rangle \notin B'$. We will use \hat{B} to find such a location. We do this using the reduction $M_{\hat{B}}^b$. Compute $l_0 = \max\langle i, j \rangle$ with $M_{\hat{B}}^0(\langle 1, i, j, x \rangle) \in A$ and $l_1 = \max\langle i, j \rangle$ with $M_{\hat{B}}^1(\langle 1, i, j, x \rangle) \in A$.
5. There are three possibilities from above:
 - If no l_0 or l_1 is found then there is no inconsistency with the 1-tableau, i.e. 1 is the right advice. If $q_1 \in A$ ACCEPT, else REJECT.
 - One l_0 or l_1 is found or $l_0 = l_1$. This means that there is an inconsistency at l_0 and 0 is the correct advice. If $q_0 \in A$ ACCEPT, else REJECT.
 - Both l_0 and l_1 are found and $l_0 \neq l_1$. This is fine, one of them must be correct and there is an inconsistency within the 1-tableau and 0 is the correct advice. If $q_0 \in A$ ACCEPT, else REJECT.

A.4 Proof of Theorem 16 - All levels

Proof. We begin with some notation: Let X^i be the set of variables $x_1^i, x_2^i, \dots, x_n^i$ and denote a particular assignment to X^i as the string ω_i where the j th bit is the value of x_j^i . For each level k , ϕ is a boolean formula over $X_1 \cup X_2 \cup \dots \cup X_k$. Now consider sets:

$$S_k(\phi) = \{X^1 \mid \forall X^2 \exists X^3 \forall \dots Q X^k \phi(X^1, X^2, \dots, X^k) \text{ is true}\}$$

$$L_k = \{\langle \phi, j \rangle \mid \text{the } j\text{-th bit in the lexicographically least } S_k(\phi) \text{ is } 1\}$$

Notice that L_k is complete for Δ_k^p [Kre88]. Let A be a $\leq_m^{p/1}$ complete set for Δ_k^p . We show how to compute L_k using a truth table reduction to A .

Suppose that M computes the reduction from L_k to A given advice b , and let $M^b(\langle \phi, j \rangle)$ be the output of the reduction, i.e. for the correct advice b it holds that $M^b(\langle \phi, j \rangle) \in A$ if and only if $\langle \phi, j \rangle \in L_k$. As above, we assume that ϕ has kn variables and we pad L_k , such that $\langle \phi, 0 \rangle, \dots, \langle \phi, n \rangle$ are all of the same length, i.e. use the same advice.

Now compute

$$\begin{aligned}\omega_0^1 &= \langle M^0\langle\phi, 0\rangle\rangle, \dots, \langle M^0\langle\phi, n\rangle\rangle \\ \omega_1^1 &= \langle M^1\langle\phi, 0\rangle\rangle, \dots, \langle M^1\langle\phi, n\rangle\rangle\end{aligned}$$

and check that either ω_0 or ω_1 makes ϕ true. At least one of these must be true as 0 or 1 must be the correct advice and return the bits of the lexicographically least assignment. This implies that the other advice either returns an assignment that does not make ϕ true, or an assignment that is not lexicographically smaller. In any case, we now have the lexicographically least assignment in hand.