Downward Translations of Equality

Christopher B. Wilson

CIS-TR-88-14
August 10, 1988

Department of Computer and Information Science
University of Oregon

Downward Translations of Equality

Christopher B. Wilson
Department of Computer and Information Science
University of Oregon
Eugene, OR 97403 USA

Abstract

The main theorem of this paper gives a general technique for constructing an oracle which separates a deterministic and nondeterministic complexity class while the classes at a higher level are equal. As a corollary of the main theorem we obtain a result in [Book, Wilson, Xu, "Relativizing Time, Space, and Time-Space," *SIAM J. Comput.*, 11(3), 1982, 571-581] that there is an oracle A such that $P^A \neq NP^A$ yet $DEXT^A = NEXT^A$. Another corollary is that for some oracle A, $DTIME^A(O(n)) \neq NTIME^A(O(n))$ and $DTIME^A(n^2) = NTIME^A(n^2)$. Some connections with sparse sets are then discussed.

1 Introduction

It is known that if $P = NP$, then $DEXT = NEXT$. This follows from a simple padding argument. In general, then, equality translates upwards (or inequality downwards). It is not known whether equality translates downwards. That is, does $DEXT = NEXT$ imply $P = NP$? In [1] there is shown the existence of an oracle set A for which $P^A \neq NP^A$ and $DEXT^A = NEXT^A$. This indicates that such a translation may not hold and, if it does, will require proof techniques which do not relativize.

In the next section we present the main theorem, which generalizes the result mentioned above. We also discuss some applications of this to results in [2], which give the best known characterizations of what $DEXT = NEXT$ implies about the P versus NP issue.

The computational models used here are deterministic and nondeterministic Turing machines equipped with the capability of making queries to an oracle. $DTIME^A(f(n))$ is defined to be the class of languages accepted by some deterministic oracle Turing machine relative to A in at most $f(n)$ steps. Note that this differs from saying $O(f(n))$, since the linear speed-up theorem does not apply to relativized computations. $NTIME^A(f(n))$ is defined similarly. If F is a class of runtime bounds, then we define $D(N)TIME^A(F)$ as $\bigcup_{f \in F} D(N)TIME^A(f(n))$.

2 Main Result and Corollaries

The main result is as follows.

Theorem 1 *If g is a monotonic function and F an enumerable class of run time bounds such that*

1. $3n \leq g(n) \leq 2^n$ *asymptotically,*
2. $\exists f \in F, f(n) \geq 2n$ *asymptotically, and*
3. $\forall f \in F, f(n) < g(\frac{n}{2}+1)+\frac{n}{2}$ *asymptotically,*

then there exists an oracle A such that $DTIME^A(F) \neq NTIME^A(F)$ and $DTIME^A(g) = NTIME^A(g)$.

proof

The oracle A will be constructed in stages. At various points, strings will be reserved (or placed into) A or $\bar{A}$. Once a string is reserved, its status will never change. Occasionally we will simulate a machine that will query the partially constructed oracle, and if an unreserved string is queried, the oracle will answer 'no' to that string.

An index $l = \langle i,j \rangle$, which describes a deterministic oracle Turing machine M_i with runtime bound $f_j \in F$, will be cancelled when we ensure that M_i^A does not accept $L(A)$ in time f_j. Initially, all indices are uncancelled. We also let NM_i be an enumeration of all nondeterministic oracle Turing machines.

In the proof we use a diagonal set $L(A)$ and a complete set $S(A)$, both based on the oracle A. They are defined as follows:

$$L(A) = \{x : \exists y, |y| = |x|, xy \in A\},$$

$$S(A) = \{\langle i,x,0^m \rangle : NM_i^A \text{ accepts } x \text{ in } \leq g(m) \text{ steps } \}.$$

We will separate the stages of the construction into odd and even stages. At

- *odd stages* we ensure that $L(A) \notin DTIME^A(F)$ and at
- *even stages* we ensure that $y \in S(A) \Leftrightarrow y0^{g(|y|)} \in A$.

Since $L(A) \in NTIME^A(F)$ for any A, the odd stages ensure that

$$DTIME^A(F) \neq NTIME^A(F).$$

Since $S(A)$ is complete for $NTIME^A(g)$, the even stages guarantee that

$$DTIME^A(g) = NTIME^A(g).$$

Construction of A

Stage 0: $A \leftarrow \emptyset$

Odd Stage $n = 2k + 1$:

Find the least uncancelled index $l = \langle i, j \rangle$. The run-time bound of M_i is $f_j \in F$. If the following four preconditions are satisfied, then we will cancel index l.

C1 $f_j(n) < g(k+1) + k + 1$

C2 $2n < g(k+1) + k + 1$

C3 n is greater than the length of the longest string queried at any earlier odd stage.

C4 $f_j(n) < 2^n$

To cancel index l, find an x ($|x| = n$) such that for no y ($|y| = |x|$) has xy been reserved. Run M_i^A on x for $f_j(n)$ steps and place all unreserved strings queried into $\bar{A}$. If M_i^A accepts x, then put all xy ($|y| = |x|$) into $\bar{A}$ (thus $x \notin L(A)$). Otherwise, put some unqueried xy ($|y| = |x|$) into A (thus $x \in L(A)$).

Even Stage $n = 2k$:

For all strings $y = \langle i, x, 0^m \rangle$, $|y| = k$, run NM_i^A on x for $g(m)$ steps. If it accepts, then put $y0^{g(k)}$ into A and reserve for $\bar{A}$ all unreserved strings queried on one accepting path.

If it rejects, then see if adding (at most $g(k)$) strings to A will force NM_i^A to accept x. If so, then reserve those strings for A, reserve $y0^{g(k)}$ for A, and reserve the remaining unreserved strings on some accepting path for $\bar{A}$. Otherwise, the behavior of NM_i^A is immune to later changes to the oracle set, so reserve $y0^{g(k)}$ for $\bar{A}$.

end construction

We now have to prove that the construction is possible.

Point 1: *An even stage does not impede the construction at the next even stage.*
At even stage $2k$, strings of length at most $g(k)$ are queried (and so reserved). At the next even stage $2(k+1)$, strings of length $g(k+1) + k + 1$ (the $y0^{g(|y|)}$) need to be unreserved.

Point 2: *At any even stage n, fewer than 2^n strings are reserved.*
When $n = 2k$, at most $g(k)$ strings are reserved by any machine represented by any of the 2^k encodings. So at most $g(k)2^k \le 2^{2k} = 2^n$ strings are reserved.

Point 3: *At an odd numbered stage $n = 2k + 1$, less than 2^n strings have been reserved by previous even stages.*
This follows from point 2 and noticing that $\sum_{i=1}^{k} 2^{2i} < 2^{2k+1} = 2^n$.

Point 4: *An odd stage does not reserve any $y0^{g(|y|)}$ which may have to be reserved at the next even stage.*
This follows from preconditions C1 and C2.

Point 5: *The construction at the even stages is possible.*
Because of points 1 and 4.

Point 6: *An odd stage does not affect the construction at the next odd stage.*
By precondition C3.

It remains to prove that the construction at the odd stages is possible. Now only the even stages cause concern, by point 6. So we must show that at stage $n = 2k+1$, we can always find some x such that for no y has xy been reserved. There are 2^n different sets $H(x) = \{xy : |x| = |y| = n\}$. By point 3, the number of strings reserved by all previous even stages is less than 2^n. Thus there must exist an x of length n for which no member of $H(x)$ is reserved.

Once such an x is found, we may still need to find some xy not queried at the end of the stage (this xy may be placed in A). The number of strings reserved at this stage is, for some j, at most $f_j(n) < 2^n = card\, H(x)$, by precondition C4. So there will always exist a y for which xy is not reserved.

Finally, note that for each index l, the preconditions C1-C4 will be satisfied infinitely often, and every index will eventually be cancelled.
□

We should point out that if F is a recursively enumerable class of recursive functions and g is a recursive function, then the oracle A is itself a recursive set. The statement of the theorem is a little technical. It does have quite a few interesting corollaries.

Corollary 2 *([1]) There is a recursive oracle A such that*

$$P^A \neq NP^A \text{ and}$$

$$DEXT^A = NEXT^A.$$

Corollary 3 *There is a recursive oracle A such that*

$$DTIME^A(2^{poly-log}) \neq NTIME^A(2^{poly-log}) \text{ and}$$

$$DEXT^A = NEXT^A.$$

Corollary 4 *There is a recursive oracle A such that*

$$DTIME^A(O(n)) \neq NTIME^A(O(n)) \text{ and}$$

$$DTIME^A(n^2) = NTIME^A(n^2).$$

Corollary 5 *For each polynomial p(n), there is a recursive oracle A such that*

$$DTIME^A(O(p(n))) \neq NTIME^A(O(p(n))) \text{ and}$$

$$DTIME^A(p(n)\log n) = NTIME^A(p(n)\log n).$$

Corollary 6 *There is a recursive oracle A such that*

$$DTIME^A(2n) \neq NTIME^A(2n) \text{ and}$$

$$DTIME^A(3n) = NTIME^A(3n).$$

In [5] there is the surprising result that $DTIME(O(n)) \neq NTIME(O(n))$ (unrelativized!). Since it is not too difficult to construct an oracle for which these two classes are equal, the result in [5] provides an excellent example of a proof technique which does not relativize. Corollary 4 seems to indicate that even given the result in [5], it will require another nonrelativizing technique to prove that deterministic and nondeterministic quadratic time differ. Thus, inequality does not smoothly translate upwards.

The well known linear speed-up theorem [3] is another result which does not relativize. This is due to the fact that while we are free to increase the size of the alphabet of a Turing machine, the oracle itself is fixed, and thus queries cannot be sped up. This is why corollary 6 appears to be anomalous.

3 Sparse Sets

In [2] it is shown that there are no sparse sets in $NP - P$ if and only if $DEXT = NEXT$. In [4] there is constructed an oracle A for which $P^A \neq NP^A$ and $NP^A - P^A$ contains no sparse sets. Since the result in [2] relativizes, the result in [4] and corollary 2 are seen to be equivalent.

Other densities also are considered in [2]. We say a set S is $s(n)$-sparse if there are at most $s(n)$ strings in S of length n. A set is thus sparse if it it $s(n)$-sparse for some polynomial $s(n)$. In addition, we sometimes need to require the strings in a sparse set to be uniformly distributed, thus defining a uniform $s(n)$-sparse set (see [2] for details). It was also shown that there are no uniform $n^{\log n}$-sparse sets in $NP - P$ if and only if $DTIME(2^{O(\sqrt{n})}) = NTIME(2^{O(\sqrt{n})})$. Theorem 1 can now be applied to yield an oracle A such that $P^A \neq NP^A$ while $NP^A - P^A$ contains no uniform $n^{\log n}$-sparse sets.

An interesting open question is to generalize the results of [2]. Perhaps it could be shown that there are no (uniform) $2^{g^{-1}(F)}$-sparse sets in $NTIME(F) - DTIME(F)$ if and only if $NTIME(g) = DTIME(g)$. The construction of theorem 1 could then be applied to construct an oracle which separates a deterministic class from a nondeterministic one in such a way that the difference of the two classes contains no sparse sets.

References

[1] R. V. Book, C. B. Wilson, and Xu M. R., "Relativizing Time, Space, and Time-Space," *SIAM J. Comput., vol. 11, no. 3*, August 1982, pp 571-581

[2] J. Hartmanis, V. Sewelson, and N. Immerman, "Sparse Sets in NP-P: EXPTIME versus NEXPTIME," *Proceedings of the 15th STOC*, 1983, pp 382-391

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

[4] S. A. Kurtz, "Sparse Sets in NP-P: Relativizations," *SIAM J. Comput.*, vol. 14, no. 1, February 1985, pp 113–119

[5] W. J. Paul, N. Pippenger, E. Szemeredi, and W. T. Trotter, "On Nondeterminism versus Determinism and Related Problems," *Proceedings of the 24th FOCS*, 1983, pp 429-438