On Parity and Near-Testability: $P^A \neq NT^A$ With Probability 1

Lane Hemachandra[*]

87-852
July 1987

Department of Computer Science
Cornell University
Ithaca, New York 14853-7501

[*] Research supported in part by NSF grant DCR-8520597 and a Fannie and John Hertz Fellowship.

On Parity and Near-Testability: $\mathrm{P}^A \neq \mathrm{NT}^A$ With Probability 1

Lane A. Hemachandra*
Department of Computer Science
Cornell University

July, 1987

Abstract

The class of near-testable sets, NT, was defined by Goldsmith, Joseph, and Young. They noted that $\mathrm{P} \subseteq \mathrm{NT} \subseteq \mathrm{PSPACE}$, and asked whether $\mathrm{P} = \mathrm{NT}$. This note shows that NT shares the same m-degree as the parity-based complexity class $\oplus\mathrm{P}$ (i.e., $\mathrm{NT} \equiv_m^p \oplus\mathrm{P}$) and uses this to prove that relative to a random oracle A, $\mathrm{P}^A \neq \mathrm{NT}^A$ with probability one. Indeed, with probability one, $\mathrm{NT}^A - (\mathrm{NP}^A \bigcup \mathrm{coNP}^A) \neq \emptyset$.

1 Introduction and Background

Definition 1.1 [GJY87a] A set S is in the class NT ("near-testable") if and only if

$$L = \{x \mid (x \in S) \oplus (x_+ \in S)\} \in \mathrm{P}.$$

Here, $\oplus$ denotes "exclusive or" and x_+ denotes the string that follows x lexicographically.

*Research supported in part by NSF grant DCR-8520597 and a Fannie and John Hertz Fellowship.

Goldsmith, Joseph, and Young ask if P equals NT, and as a partial answer show that if one-way functions exist (equivalently, if P $\neq$ UP, see [GS84]), then P $\neq$ NT [GJY87a].

In their proof they identify parity as a powerful tool for dealing with the class NT. This note goes further and suggests that parity is not only a tool, but is the answer to the question, "Where does NT fall in the scheme of standard complexity classes?"

Section 2 pinpoints the location of NT by proving that NT is many-one polynomial-time equivalent to the standard complexity class $\oplus$P of Papadimitriou and Zachos [PZ82,PZ83]. Since UP $\subseteq \oplus$P, the UP result of [GJY87a] follows as an immediate corollary.

Section 3 notes that versions of near-testability defined in far more general ways remain subsets of, and many-one equivalent to, $\oplus$P.

Section 4 notes that $\mathrm{P}^A \neq \mathrm{NT}^A$ with probability one relative to a random oracle A. This says that in almost every relativized world, NT and P differ. Indeed, we show stronger probability one results for NT: with probability one NT^A contains sets not in NP^A, coNP^A, or even PP^A. These results are consequences of the probability one techniques of Bennett and Gill [BG81], and of the fact that NT and $\oplus$P share an m-degree.

2 $\oplus\mathrm{P} \equiv_m^p \mathrm{NT}$

2.1 $\oplus$P

The class $\oplus$P, "parity P," is the class of languages that determine the parity of the number of accepting paths of nondeterministic polynomial-time Turing machines.

Definition 2.1 [PZ82,PZ83] $\oplus\mathrm{P} = \{L \mid$ there is a nondeterministic polynomial-time Turing machine N_i such that $[x \in L \iff N_i(x)$ has an odd number of accepting paths$]\}$.

Papadimitriou and Zachos show that $\oplus\mathrm{P}^{\oplus\mathrm{P}} = \oplus\mathrm{P}$, and thus $\oplus$P has behavior that seems to differ from that of NP.

It is easy to note that:

Lemma 2.2 $\mathrm{P} \subseteq \mathrm{UP} \subseteq \oplus\mathrm{P} \subseteq \mathrm{P}^{\#\mathrm{P}[1]} \subseteq \mathrm{P}^{\#\mathrm{P}}$, where [1] indicates that on any input only one oracle call is made.

UP [Val76,GS84,HH86] is Valiant's uniqueness class and #P [Val79a,Val79b] is Valiant's class of counting functions.

Proof: UP $\subseteq \oplus$P as a UP machine for a language L instantly (since 0 is even and 1 is odd) provides the machine N_i required by Definition 2.1 to prove that L is in $\oplus$P. The other inclusions are immediate.

♠

2.2 $\oplus$P and NT have the same m-degree

We say that A is many-one polynomial-time reducible to B ($A \leq_m^p B$) if there is a polynomial-time computable function f so that for all strings x, $x \in A \iff f(x) \in B$ [GJ79]. An m-degree is an equivalence class of sets with respect to many-one polynomial-time reductions (see, e.g., [KMR86]). This section shows that NT and $\oplus$P share the same m-degree, and that NT $\subseteq \oplus$P.

Theorem 2.3 NT $\subseteq \oplus$P.

Lemma 2.4 $\oplus$P $\leq_m^p$ NT.

Theorem 2.5 NT $\equiv_m^p \oplus$P.

Proof of Theorem 2.5 The theorem follows immediately from Lemma 2.4 and Theorem 2.3.

♠

Proof of Theorem 2.3 Let $L \in$ NT. Let polynomial-time language L' do the testing, i.e.,

$$x \in L' \iff \Big((x \in L) \oplus (x_+ \in L)\Big).$$

Let N_L by the nondeterministic polynomial-time Turing machine that on input x spawns, for each string y that is lexicographically less that x, a path that accepts if and only if $y \in L'$. Also, if the lexicographically first string, ϵ, is in L (this information is coded into N_L), then let N_L always have one additional path that mindlessly accepts. Now $L \in \oplus$P, taking machine N_L to be the machine N_i of Definition 2.1.

♠

Proof of Lemma 2.4 Suppose $L \in \oplus\mathrm{P}$, and let N_i be the machine (of Definition 2.1) that whose paths certify that $L \in \oplus\mathrm{P}$. We formalize a "path" as a zero-one vector that contains the nondeterministically "guessed" bits. Note that we can easily modify machine N_i to create a machine N_j such that

1. N_j certifies $L \in \oplus\mathrm{P}$ (i.e., $x \in L \iff N_j(x)$ has an odd number of accepting paths),

2. N_j runs (for some fixed k that depends on L) in $\mathrm{NTIME}[n^{k+1} + k]$, and

3. machine $N_j(x)$ starts by nondeterministically guessing an $|x|^k$ bit guess vector, and then (each of the $2^{|x|^k}$ paths) proceeds deterministically.

•

Let $L' = \{x\#\mathit{path} \mid \ |\mathit{path}| = |x|^k$ and there are an odd number of accepting paths of $N_j(x)$ that are lexicographically $\leq \mathit{path}\}$.

Crucially, L' is in NT as (for all paths except the lexicographically first, which is an easy case to handle):

$$[(x\#\mathit{path} \in L') \oplus (x\#\mathit{path}_- \in L')] \iff \mathit{path}_- \text{ is an accepting path of } N_j(x),$$

where path_- indicates the path lexicographically preceding path. Also, $L \leq_m^p \oplus\mathrm{P}$: we reduce "$x \in L$?" to "$x\#1^{n^k} \in L'$?" This works as 1^{n^k} is the lexicographically last path on input x, and by the definition of L', $x\#1^{n^k}$ is in L' exactly when $N_j(x)$ has an odd number of accepting paths.

Thus we have many-one reduced a general language L in $\oplus\mathrm{P}$ to a language L' in NT. So $\oplus\mathrm{P} \leq_m^p$ NT.

♠

Corollary 2.6 $\mathrm{P} = \mathrm{NT}$ if and only if $\mathrm{P} = \oplus\mathrm{P}$.

As a consequence, we immediately know the effect of structural assumptions about classes bigger or smaller than $\oplus\mathrm{P}$ on the P=NT question. For example, by Lemma 2.2, we can conclude that $\mathrm{P} \neq \mathrm{UP} \Rightarrow \mathrm{P} \neq \mathrm{NT}$ [GJY87a], and $\mathrm{P} = \mathrm{P}^{\#\mathrm{P}} \Rightarrow \mathrm{P} = \mathrm{NT}$. However, Theorem 2.5 is a more general and powerful locator of the position and structure of $\oplus\mathrm{P}$, and thus forms our stepping stone for the probability one results of the next section.

It is routine to verify that the results of this section relativize.

Definition 2.7 $\oplus\mathrm{P}^A = \{L \mid$ there is a nondeterministic polynomial-time Turing machine N_i such that $[x \in L \iff N_i^A(x)$ has an odd number of accepting paths$]\}$.

Definition 2.8 A set S is in NT^A if and only if

$$L = \{x \mid (x \in S) \oplus (x_+ \in S)\} \in \mathrm{P}^A.$$

Theorem 2.9 For all oracles A, $\mathrm{NT}^A \subseteq \oplus\mathrm{P}^A$.

Lemma 2.10 For all oracles A, $\oplus\mathrm{P}^A \leq_m^p \mathrm{NT}^A$.

Theorem 2.11 For all oracles A, $\mathrm{NT}^A \equiv_m^p \oplus\mathrm{P}^A$.

3 Generalizing NT

Goldsmith, Joseph, and Young suggest the possibility of a more general notion of near-testability [GJY87b]. We show that their notion, and far more general notions of near-testability, are still subsets of $\oplus\mathrm{P}$.

Definition 3.1 (See [Ko83,GJY87b] for related ideas.)

1. A total[1] ordering $\prec$ on Σ^* is polynomially well-founded and exponentially length related if there is a polynomial $p()$ and an exponential function $e()$ (i.e., for some k, $e(k) = O(2^{n^k})$) such that:
   (a) $y \prec x?$ is testable in $\oplus\mathrm{P}$ (i.e., $\{(y, x) \mid y \prec x\} \in \oplus\mathrm{P}$),
   (b) $x \prec y$ implies that $|x| \leq p(|y|)$,
   (c) the length of a $\prec$-descending chain is shorter than e of the length of its maximal element, and
   (d) $(\forall z \in \Sigma^* - \epsilon)[\epsilon \prec z]$.

[1] In fact, Theorem 3.4 would hold even if we allowed our order to be a tree-like partial order rooted at ϵ.

2. We call such an ordering a *nice* ordering.

Note that NT is defined using standard lexicographical order, which is a common example of a nice ordering. The new class NewT defined below is defined in a quite general way. Nonetheless, like NT, NewT is a subset of $\oplus$P.

Definition 3.2 A set S belongs to the class NewT if there is a nice order $\prec$ such that $L = \{x \mid (x \in S) \oplus (x_+ \in S)\} \in \oplus\mathrm{P}$. Here, x_+ denotes the immediate successor of x in our well-founded linear ordering $\prec$.

Note that this is a strong generalization of NT. We allow a general ordering, for which $\prec$ may not even be testable in polynomial time, and our "xor" language itself, L above, may not be computable in polynomial time. Both are allowed to be $\oplus$P computations. If both were restricted to P computations (call the resulting class NewT′), we'd have the extension of NT suggested in [GJY87b].

Lemma 3.3 NT $\subseteq$ NewT′ $\subseteq$ NewT.

Theorem 3.4 NewT $\subseteq \oplus$P.

Corollary 3.5 $\oplus\mathrm{P} \equiv_m^p \mathrm{NewT} \equiv_m^p \mathrm{NewT}' \equiv_m^p \mathrm{NT}$. That is, $\oplus$P, NewT, NewT′, and NT have the same m-degree.

Proof Lemma 3.3: Immediate from the definitions.

♠

Proof of Corollary 3.5 Immediate from Lemma 3.3, Theorem 3.4, and Lemma 2.4.

♠

Proof of Theorem 3.4: Our proof extends, but shares the spirit of, the proof of Theorem 2.3. However, we must account carefully for the action of the $\oplus$P computations that are now allowed as part of the NewT definition.

Assume L is an arbitrary language in NewT. We will show that $L \in \oplus$P. We'll use the term *$\oplus$P machine* to denote a nondeterministic polynomial-time Turing machine operating under the $\oplus$P acceptance mechanism—that is, the machine is considered to accept if and only if it has an odd number of accepting paths.

Let $\prec$ be the ordering from the definition of NewT, let N_1 be the $\oplus$P machine accepting $\{(y, x) \mid y \prec x\}$, and let N_2 be the $\oplus$P machine accepting $\{x \mid (x \in L) \oplus (x_+ \in L)\}$.

Without loss of generality, assume $\epsilon \notin L$. (If $\epsilon \in L$, the same proof works, except we add a dummy accepting path to the machine N_4 (below) to flip its parity.)

Let N_3 be the $\oplus$P machine that on input (a, b) starts simulating $N_1(a, b)$ but on each path of $N_1(a, b)$ that is about to accept, $N_3(a, b)$ instead of accepting simulates $N_2(a)$.

Finally, here is the $\oplus$P machine, N_4, that accepts L. On input x, N_4 nondeterministically makes a path, $path_y$, for each string y such that $|y| \leq p(|x|)$, where p is the polynomial bound on the length-relatedness of the nice ordering $\prec$. On $path_y$, simulate $N_3(y, x)$.

Correctness: If $y \not\prec x$ then $N_1(y, x)$ has an even number of accepting paths, so regardless of whether $N_2(y)$ has an even or an odd number of paths, $N_3(y, x)$ will have an even number of paths and will not change the parity of $N_4(x)$. On the other hand, if $y \prec x$, then $N_3(y, x)$ will accept (i.e., have an odd number of accepting paths) if and only if $(y \in L) \oplus (y_+ \in L)$. Since we guess all y along the unique maximal chain from x to ϵ, and $\epsilon \notin L$, we have $x \in L$ if and only if $N_4(x)$ has an odd number of accepting paths (i.e., ϵ was not in L, and an odd number of times along the chain we switched between being in and out (or out and in) of L).

♠

4 Probability 1 Results for NT

Bennett and Gill [BG81] began the study of what happens when complexity classes are relativized with a random oracle. A stream of extensions and related work has followed their seminal paper [Cai86b,Cai86a,Har85,Kur82].

In this section, we note that the characterization of NT developed in the previous section, combined with the proof methods of [BG81], shows that with probability one, NT^A contains sets computationally hard sets. Indeed, with probability one, NT^A contains sets that are neither in NP^A nor in coNP^A.

Lemma 4.1 Relative to a random oracle A, $\oplus\mathrm{P}^A - \mathrm{PP}^A \neq \emptyset$ with probability one.

Theorem 4.2 Relative to a random oracle A, $\mathrm{NT}^A - \mathrm{PP}^A \neq \emptyset$ with probability one.

Corollary 4.3 Relative to a random oracle A, $\mathrm{NT}^A - (\mathrm{NP}^A \bigcup \mathrm{coNP}^A) \neq \emptyset$ with probability one.

Corollary 4.4 Relative to a random oracle A, $\mathrm{NT}^A \supsetneq \mathrm{P}^A$ with probability one.

Proof of Lemma 4.1 Theorem 3 of [BG81, p. 103] shows that $\mathrm{PP}^A \subsetneq \mathrm{PSPACE}^A$ with probability one. However, their proof in fact uses a parity based language that not only is in PSPACE^A, but also is easily seen to be in $\oplus\mathrm{P}^A$. Thus, the proof of their Theorem 3 also proves the stronger statement of our Lemma 4.1.

♠

Proof of Lemma 4.2 Let B we an oracle for which $\oplus\mathrm{P}^B - \mathrm{PP}^B \neq \emptyset$, and suppose L is a language in $\oplus\mathrm{P}^B - \mathrm{PP}^B$. By Lemma 2.10, there is a language $L' \in \mathrm{NT}^B$ so $L \leq_m^p L'$. Since probabilistic polynomial time is closed downwards under many-one reductions, it follows that $L' \notin \mathrm{PP}^B$, thus $\mathrm{NT}^B - \mathrm{PP}^B \neq \emptyset$. It follows from this and Lemma 4.1 that for a random oracle A, $\mathrm{NT}^A - \mathrm{PP}^A \neq \emptyset$ with probability one.

♠

Proofs of Corollaries 4.3 and 4.4 Corollary 4.3 follows directly from Theorem 4.2 and the fact that, for every oracle B, $\mathrm{PP}^B \supseteq (\mathrm{NP}^B \bigcup \mathrm{coNP}^B)$. Corollary 4.4 follows from Corollary 4.3.

♠

Thus we have shown that, with probability one, NT contains hard languages.

Comment The proceeding theorems show that for a random oracle A, there are languages in NT^A that are not in $\mathrm{NP}^A \bigcup \mathrm{coNP}^A$ with probability one. Looking for a contrasting result, we can show by direct diagonalization that there are relativized worlds B in which both NP^B and coNP^B contain sets that are not in NT^B.

Theorem 4.5 There is an oracle B such that $\mathrm{NP}^B - \mathrm{NT}^B \neq \emptyset$ and $\mathrm{coNP}^B - \mathrm{NT}^B \neq \emptyset$. [2]

5 Summary

We noted that the class NT shares an m-degree with $\oplus$P, and used this to prove that with probability one relative to a random oracle, NT^A contains computationally hard languages.

6 Acknowledgements

I thank Professor J. Hartmanis and M. Novick for helpful comments.

[2] **Proof Sketch** For the NP case, set $L = \{1^n \mid (\exists y, z)[|y| = n \wedge y \in B \wedge y = 1z]\} \in \mathrm{NP}^B$. By direct diagonalization (against the possible polynomial-time testing machines), insure $L \notin \mathrm{NT}^B$. To knock out a testing machine, run it on 1^m for m much larger than anything used in previous stages; whatever it replies make it wrong (if needed, toss a length m string that was not touched in the run into oracle B).

For the coNP case, $\overline{L}$ is in coNP^B and $\overline{L} \notin \mathrm{NT}^B$ (as NT^B is closed under complement). ♠

References

[BG81] C. Bennett and J. Gill. Relative to a random oracle A, $P^A \neq NP^A$ with probability 1. *SIAM J. on Computing*, 10:96–113, 1981.

[Cai86a] J. Cai. *On Some Most Probable Separations of Complexity Classes.* PhD thesis, Cornell University, Ithaca, NY, 1986.

[Cai86b] J. Cai. With probability one, a random oracle separates PSPACE from the polynomial-time hierarchy. In *18th ACM Symposium on Theory of Computing*, pages 21–29, 1986.

[GJ79] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman and Company, 1979.

[GJY87a] J. Goldsmith, D. Joseph, and P. Young. Self-reducible, P-selective, near-testable, and P-cheatable sets: the effect of internal structure on the complexity of a set. In *Proceedings 2nd Structure in Complexity Theory Conference*, pages 50–59, 1987.

[GJY87b] J. Goldsmith, D. Joseph, and P. Young. *Self-Reducible, P-Selective, Near-Testable, and P-Cheatable Sets: The Effect of Internal Structure on the Complexity of a Set.* Technical Report 87-06-02, University of Washington, Seattle, WA, June 1987.

[GS84] J. Grollmann and A. Selman. Complexity measures for public-key cryptosystems. In *Proceedings 25th IEEE Symposium on Foundations of Computer Science*, pages 495–503, 1984.

[Har85] J. Hartmanis. Solvable problems with conflicting relativizations. *Bulletin of the European Association for Theoretical Computer Science*, 27:40–49, October 1985.

[HH86] J. Hartmanis and L. Hemachandra. Complexity classes without machines: on complete languages for UP. In *Automata, Languages, and Programming (ICALP 1986)*, pages 123–135, Springer-Verlag *Lecture Notes in Computer Science #226*, July 1986.

[KMR86] S. Kurtz, S. Mahaney, and J. Royer. Collapsing degrees. In *Proceedings 27th IEEE Symposium on Foundations of Computer Science*, pages 380–389, 1986.

[Ko83] K. Ko. On self-reducibility and weak P-selectivity. *Journal of Computer and System Sciences*, 26:209–221, 1983.

[Kur82] S. A. Kurtz. On the random oracle hypothesis. In *14th ACM Symposium on Theory of Computing*, pages 224–230, 1982.

[PZ82] C. Papadimitriou and S. Zachos. *Two Remarks on the Power of Counting.* Technical Report MIT/LCS/TM-228, Laboratory for Computer Science, MIT, Cambridge, MA, August 1982.

[PZ83] C. Papadimitriou and S. Zachos. Two remarks on the power of counting. In *Proceedings 6th GI Conference on Theoretical Computer Science*, pages 269–276, Springer-Verlag Lecture Notes in Computer Science #145, 1983.

[Val76] L. Valiant. The relative complexity of checking and evaluating. *Information Processing Letters*, 5:20–23, 1976.

[Val79a] L. Valiant. The complexity of computing the permanent. *Theoretical Computer Science*, 8:189–201, 1979.

[Val79b] L. Valiant. The complexity of enumeration and reliability problems. *SIAM Journal on Computing*, 8(3):410–421, 1979.