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: 
$P^A \neq NT^A$ 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 $P \subseteq NT \subseteq PSPACE$, and asked whether $P = NT$. This note shows that $NT$ shares the same $m$-degree as the parity-based complexity class $\oplus P$ (i.e., $NT \equiv^P_m \oplus P$) and uses this to prove that relative to a random oracle $A$, $P^A \neq NT^A$ with probability one. Indeed, with probability one, $NT^A - (NP^A \cup coNP^A) \neq \emptyset$.

1 Introduction and Background

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

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

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 \( P^A \neq NT^A \) 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.

\[ 2 \quad \oplus P \equiv_p^m NT \]

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

Papadimitriou and Zachos show that \( \oplus P^P = \oplus P \), and thus \( \oplus P \) has behavior that seems to differ from that of NP.

It is easy to note that:

**Lemma 2.2** \( P \subseteq UP \subseteq \oplus P \subseteq P^{\#P[1]} \subseteq P^{\#P} \), 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.

\(\spadesuit\)

### 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.

\(\spadesuit\)

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

\[ x \in L' \iff \left( (x \in L) \oplus (x_+ \in L) \right). \]

Let \(N_L\) by the nondeterministic polynomial-time Turing machine that on input \(x\) spawns, for each string \(y\) that is lexicographically less than \(x\), a path that accepts if and only if \(y \in L'\). Also, if the lexicographically first string, \(\epsilon\), 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.

\(\spadesuit\)
**Proof of Lemma 2.4** Suppose $L \in \oplus P$, and let $N_i$ be the machine (of Definition 2.1) that whose paths certify that $L \in \oplus P$. 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 $L \in \oplus P$ (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 $\text{NTIME}[n^{k+1} + k]$, 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 $L' = \{x \# \text{path} \mid |\text{path}| = |x|^k$ and there are an odd number of accepting paths of $N_j(x)$ that are lexicographically $\leq \text{path}\}$.

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

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

where $\text{path}_-$ indicates the path lexicographically preceding $\text{path}$. Also, $L \leq_m \oplus P$: we reduce "$x \in L""$ to "$x \# 1^n \in L'"". This works as $1^n$ is the lexicographically last path on input $x$, and by the definition of $L'$, $x \# 1^n$ 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 $\oplus P$ to a language $L'$ in NT. So $\oplus P \leq_m \text{NT}$.

\hfill \spadesuit

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

As a consequence, we immediately know the effect of structural assumptions about classes bigger or smaller than $\oplus P$ on the $P=\text{NT}$ question. For example, by Lemma 2.2, we can conclude that $P \neq \text{UP} \Rightarrow P \neq \text{NT}$ [GJY87a], and $P = P^{\# P} \Rightarrow P = \text{NT}$. However, Theorem 2.5 is a more
general and powerful locator of the position and structure of $\oplus P$, 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** \( \oplus P^A = \{ L \mid \text{there is a nondeterministic polynomial-time Turing machine } N_i \text{ such that } x \in L \iff N_i^A(x) \text{ has an odd number of accepting paths}\}. \)

**Definition 2.8** A set \( S \) is in NT\(^A\) if and only if

\[
L = \{ x \mid \text{(} x \in S \text{)} \oplus \text{(} x^+ \in S \text{)} \} \in \text{P}^A.
\]

**Theorem 2.9** For all oracles \( A \), NT\(^A\) \( \subseteq \oplus \text{P}^A \).

**Lemma 2.10** For all oracles \( A \), \( \oplus \text{P}^A \leq_m \text{NT}^A \).

**Theorem 2.11** For all oracles \( A \), NT\(^A\) \( \equiv_m \oplus \text{P}^A \).

## 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 \( \oplus \text{P} \).

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

1. A total\(^1\) ordering \( \prec \) on \( \Sigma^* \) 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 \( \oplus \text{P} \) (i.e., \( \{(y, x) \mid y \prec x \} \in \oplus \text{P} \)),

(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) \( (\forall z \in \Sigma^* - e)[e \prec z] \).

---

\(^1\)In fact, Theorem 3.4 would hold even if we allowed our order to be a tree-like partial order rooted at \( e \).
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 $L = \{x \mid (x \in S) \oplus (x_+ \in S)\} \in \oplus P$. 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** $\oplus P \equiv_m^p$ NewT $\equiv_m^p$ NewT' $\equiv_m^p$ NT. That is, $\oplus P$, NewT, NewT', and NT have the same $m$-degree.

**Proof Lemma 3.3:** Immediate from the definitions.

博文

**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 \( \epsilon \not\in L \). (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 \( y \neq x \) 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 \( \epsilon \), and \( \epsilon \not\in L \), we have \( x \in L \) if and only if \( N_4(x) \) has an odd number of accepting paths (i.e., \( \epsilon \) 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 \( L \).

\( \blacklozenge \)

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 $A$, $\oplus P^A - PP^A \neq \emptyset$ with probability one.

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

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

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

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

Proof of Lemma 4.2 Let $B$ be an oracle for which $\oplus P^B - PP^B \neq \emptyset$, and suppose $L$ is a language in $\oplus P^B - PP^B$. By Lemma 2.10, there is a language $L' \in NT^B$ so $L \leq^p_m L'$. Since probabilistic polynomial time is closed downwards under many-one reductions, it follows that $L' \not\subseteq PP^B$, thus $NT^B - PP^B \neq \emptyset$. It follows from this and Lemma 4.1 that for a random oracle $A$, $NT^A - PP^A \neq \emptyset$ 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 $B$, $PP^B \supseteq (NP^B \cup coNP^B)$. 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 $NP^A \cup coNP^A$ 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 $\text{NP}^B - \text{NT}^B \neq \emptyset$ and $\text{coNP}^B - \text{NT}^B \neq \emptyset$.  

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.

---

2Proof Sketch For the NP case, set $L = \{1^n \mid (\exists y, z)[|y| = n \land y \in B \land y = 1z]\} \in \text{NP}^B$. By direct diagonalization (against the possible polynomial-time testing machines), insure $L \not\in \text{NT}^B$. 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 $\overline{L} \not\in \text{NT}^B$ (as NT$^B$ is closed under complement).

$\diamondsuit$
References


Springer-Verlag Lecture Notes in Computer Science #226, July 1986.


