On Randomized Versus Deterministic Computation

Marek Karpinski*
Dept. of Computer Science
University of Bonn
5300 Bonn 1
and
International Computer Science Institute
Berkeley, California

Rutger Verbeek †
Dept. of Computer Science
FernUniversität Hagen
5800 Hagen 1

Abstract

In contrast to deterministic or nondeterministic computation, it is a fundamental open problem in randomized computation how to separate different randomized time classes (at this point we do not even know how to separate linear randomized time from $O(n^{\log n})$ randomized time) or how to compare them relative to corresponding deterministic time classes. In another words we are far from understanding the power of *random coin tosses* in the computation, and the possible ways of simulating them deterministically.

In this paper we study the relative power of linear and polynomial randomized time compared with exponential deterministic time. Surprisingly, we are able to construct an oracle A such that exponential time (with or without the oracle A) is simulated by linear time Las Vegas algorithms using the oracle A. For Las Vegas polynomial time (ZPP) this will mean the following equalities of the time classes:

$$\mathrm{ZPP}^A = \mathrm{EXPTIME}^A = \mathrm{EXPTIME}(= \mathrm{DTIME}(2^{poly})) \; .$$

Furthermore, for all the sets

$$M \subseteq \Sigma^* \; : \; M \leq_{UR} \bar{A} \Longleftrightarrow M \in \mathrm{EXPTIME}$$

*Supported in part by the Leibniz Center for Research in Computer Science, by the DFG Grant KA 673/4-1 and by the SERC Grant GR-E 68297. Email: marek@cs.uni-bonn.de.

†Part of the research was done while visiting the International Computer Science Institute, Berkeley, California. Email: verbeek@fernuni-hagen.de.

($\leq_{UR}$ being unfaithful polynomial random reduction, c.f. [Jo 90]).

Thus $\bar{A}$ is $\leq_{UR}$ complete for EXPTIME, but interestingly not NP-hard under (deterministic) polynomial reduction unless EXPTIME=NEXPTIME. We are also able to prove, for the first time, that randomized reductions are exponentially more powerful than deterministic or nondeterministic ones (cf. [AM 77]). Moreover, a set B is constructed such that Monte Carlo polynomial time (BPP) under the oracle B is exponentially more powerful than deterministic time with nondeterministic oracles, more precisely:

$$\mathrm{BPP}^B = \Delta_2\mathrm{EXPTIME}^B = \Delta_2\mathrm{EXPTIME}(= \mathrm{DTIME}(2^{poly})^{\mathrm{NTIME}(n)}) \ .$$

This strengthens considerably a result of Stockmeyer [St 85] about the polynomial time hierarchy that for some decidable oracle B, $\mathrm{BPP}^B \not\subseteq \Delta_2\mathrm{P}^B$. Under our oracle BPP^B is exponentially more powerful than $\Delta_2 P^B$, and B does not add any power to Δ_2EXPTIME.

1 Randomized Computation

A probabilistic Turing machine (PTM) is a standard Turing machine with the ability to *toss* a *random coin*, and can be viewed as a nondeterministic machine with a different accepting condition: an input $x \in \Sigma^*$ is accepted (in time $T(n)$) if more than a half of the computations (of length $T(|x|)$) are accepting.

The probability of accepting (rejecting) can be defined as the fraction of accepting (rejecting) paths in the normalized computation tree (i.e. all the paths have the same number of binary branching points). We will restrict ourselves to machines with a clock: all computations have the length at most $T(|x|)$.

We shall study the following classes of probabilistic (bounded error) Turing machines:

- **Monte Carlo machines** (Bounded error PTMs, **MTM**s)
  any input is accepted either with probability $> \frac{3}{4}$ or with probability $< \frac{1}{4}$.
- **Randomized machines** (one sided error PTMs, **RTM**s):
  any input is accepted with probability $> \frac{3}{4}$ or 0.
- **Las Vegas machines** (zero error PTMs, **ZPTM**s):
  either x is accepted with probability $> \frac{3}{4}$ and rejected with probability 0 or x is rejected with probability $> \frac{3}{4}$ and accepted with probability 0.

We denote the corresponding complexity classes by

$$\mathrm{PrTIME}(T) \ = \ \{L(\mathcal{M}) \mid \mathcal{M} \text{ is an } O(T)\text{-bounded PTM}\}$$

$\mathrm{BPTIME}(T)$ (same for MTMs)

$\mathrm{RTIME}(T)$ (same for RTMs)

$\mathrm{ZPTIME}(T)$ (same for ZPTMs)

Other than in the deterministic case it is not clear that the "linear speed up" is valid for Monte Carlo, Randomized, and Las Vegas machines.

The polynomial time classes are denoted as usual by

$$\mathrm{PP}\ (= \bigcup_k \mathrm{PrTIME}(n^k)),\ \mathrm{BPP},\ \mathrm{RP},\ \textit{and}\ \mathrm{ZPP}.$$

All these machines can be relativized in a canonical way. The relativized machines, sets, complexity classes (with oracle A) are (as usual) denoted by $\mathcal{M}^A$, $L(\mathcal{M}^A)$, e.g. BPP^A; if C is a set of oracle sets, the union of relativized classes with oracle $A \in C$ is denoted by superscript C (e.g. $\mathrm{BPP}^{NP} = \bigcup_{A \in NP} \mathrm{BPP}^A$).

Other than deterministic or nondeterministic machines, PTMs with bounded error (MTMs, RTMs, or ZPTMs) cannot be described by the syntactical properties only. The MTMs (RTMs, ZPTMs) form nonenumerable subsets of the PTMs. Thus ZPP, RP and BPP have probably no complete sets. Therefore, we do not have any method for proving that $\mathrm{BPTIME}(n) \neq \mathrm{BPTIME}(n^{\log n})$ [KV 88] and we cannot exclude the situation that (at least under some oracle) $\mathrm{ZPTIME}(n) = \mathrm{BPP}$. In [FS 89] the existence of such an oracle is claimed but unfortunately the construction used in the proof seems to have an irreparable flow [F 92]. The paper [FS 89] was also a starting point of our investigation.

A related notion of a probabilistic Turing machine with an oracle was introduced recently by A. Yao in a context of program checkers [Y 90].

Under the random oracle BPP (and RP, ZPP) equals P and reasonable hierarchy theorems are valid ([BG 81]). Most researchers believe that the power of ZPP does not (or not by much) exceed P. BPP is included in Σ_2^P and thus in the polynomial hierarchy. On the other hand, under some oracle, $\mathrm{BPP} \not\subseteq \Delta_2^P$ [St 85]. We will show that under appropriate oracles ZPP=EXPTIME and $\mathrm{BPP} = \Delta_2\mathrm{EXPTIME}$. This means: under some oracle the zero-error PTMs are exponentially more powerful that their deterministic counterparts, and bounded error PTMs are exponentially more powerful than nondeterministic machines.

The results have also consequences for the unrelativized world: we can show that the Las Vegas reductions are exponentially more powerful than deterministic reductions, and the Monte Carlo reductions are exponentially more powerful than γ-reductions.

While the definition of the polynomial hierarchy in q we will need a generalization of the

well known polynomial hierarchy (in a relativized version):

$$\begin{aligned}
\Sigma_0\mathrm{TIME}(T)^A &= \Pi_0\mathrm{TIME}(T)^A = \Delta_0\mathrm{TIME}(T)^A = \mathrm{DTIME}(T)^A \\
\Delta_{n+1}\mathrm{TIME}(T)^A &= \mathrm{DTIME}(T)^{\Sigma_n\mathrm{TIME}_{(n)^A}} \\
\Sigma_{n+1}\mathrm{TIME}(T)^A &= \mathrm{NTIME}(T)^{\Sigma_n\mathrm{TIME}_{(n)^A}} \\
\Pi_{n+1}\mathrm{TIME}(T)^A &= \text{co-NTIME}(T)^{\Sigma_n\mathrm{TIME}_{(n)^A}} = \{\Sigma^* \setminus A \mid A \in \Sigma_n\mathrm{TIME}(T)^A\}\,.
\end{aligned}$$

To avoid confusion with oracle classes we prefer $\Sigma_k P$ etc. for the classes of the polynomial hierarchy $\Sigma_k^P = \bigcup_i \Sigma_k\mathrm{TIME}(n^i)$; e.g. $NP = \Sigma_1 P$. It is easy to see that for all at least linearly increasing T

$$\Sigma_{n+1}\mathrm{TIME}(T)^A \supseteq \mathrm{NTIME}(n)^{\Sigma_n\mathrm{TIME}_{(T)^A}}$$

and this inclusion is strict for some oracle A.

Let EXTIME denote $\bigcup_k \mathrm{DTIME}(2^{kn})$, and let NEXTIME, BPEXTIME, Σ_kEXTIME, etc. denote the other exponential time classes. In the same way let EXPTIME(NEXPTIMEetc.) denote $\bigcup_k \mathrm{DTIME}(2^{n^k})$ ($\bigcup_k \mathrm{NTIME}(2^{n^k})$ etc.).

2 Oracle A with $\mathbf{ZPP}^A = \mathbf{EXPTIME}^A = \mathbf{EXPTIME}$

We will construct an oracle A such that for all deterministic oracle machines $\mathcal{M}_i$ running in time 2^n and all $x \in \Sigma^*$ with $|x| = n > i$

$$\begin{aligned}
x \notin L(\mathcal{M}_i^A) &\implies \forall \rho \in \Sigma^{4n}\,, < i, x, \rho > \notin A \\
x \in L(\mathcal{M}_i^A) &\implies \#\{\rho \in \Sigma^{4n} \mid < i, x, \rho > \in A\} > \frac{3}{4} \cdot 2^{4n}
\end{aligned}$$

This set A has the property

$$\mathrm{DTIME}(2^n)^A \subseteq \mathrm{DTIME}(2^{6n})\,.$$

By standard padding arguments we can conclude

Theorem 1. *There exists an oracle A, such that*

$$\begin{aligned}
ZPTIME(n)^A = EXTIME^A = EXTIME\,, \\
ZPP^A = EXPTIME^A = EXPTIME\,.
\end{aligned}$$

The (surprisingly simple) construction uses the fact, that deterministic exponential time machines cannot query all oracle strings of linear length.

First of all some notations:

- $< i, x, \rho >$ will denote the string $\$^i x \$ \rho$. The oracles will be subsets of $\{0, 1, \$\}^* = (\Sigma \cup \{\$\})^*$.
- The following ordering of pairs $(i, x) \in \mathbb{N} \times \not{\kappa}^*$ is used:
  $(i, x) < (j, y)$ if one of the following holds
  (1) $|x| < |y|$
  (2) $|x| = |y|$ and $x < y$ (lexicographically)
  (3) $x = y$ and $i < j$.
- (Restricted to pairs (i, x) with $i < |x|$ this is a linear ordering of (ordinal) type ω.)

Without loss of generality we restrict ourselves to the input alphabet $\Sigma = \{0, 1\}$.

Construction of the Oracle A

A is constructed in stages following the above defined ordering. The (initially empty) set A is augmented during the construction at stage (i, x) by strings $< i, x, \rho >$, when $x \in L(\mathcal{M}_i^A)$. Queries "$< i, x, \rho > \in A$?" on previous stages are recorded in a set D; these are not changed when "$x \in L(\mathcal{M}_i^A)$" is encoded.

Stage $(0, \epsilon)$: $A := \emptyset$; $D := \emptyset$.

Stage (i, x), $i < |x|$:
Simulate at most $2^{|x|}$ steps of $\mathcal{M}_i^A(x)$.
If $\mathcal{M}_i^A$ asks "$< j, y, \rho > \in A$?", $(j, y) > (i, x)$, and $|\rho| = 4 \cdot |y|$,
then $D := D \cup \{< j, y, \rho >\}$ (i.e. $< j, y, \rho > \notin A$ is fixed).
If $\mathcal{M}_i^A$ accepts x, then $A := A \cup (\{< i, x, \rho > \mid |\rho| = 4 \cdot |x|\} \setminus D)$. □

Lemma 1. *For all i, x $(i < |x|)$ the following holds:*

(1) If $\mathcal{M}_i^A$ accepts $x \in \Sigma^n$ within 2^n steps, then $< i, x, \rho > \notin A$ for at most $2n \cdot 2^{2n}$ strings $< i, x, \rho >$ with $|\rho| = 4n$.

(2) If $\mathcal{M}_i^A$ does not accept $x \in \Sigma^n$ within 2^n steps, then $< i, x, \rho > \notin A$ for all ρ.

Proof. Though the oracle is changed during the construction, all oracle queries are answered consistently. A new string is added to A only if it was not queried during previous steps.

(1) If $\mathcal{M}_i^A$ accepts $x \in \Sigma^n$ within the time bound, all strings $< i, x, \rho >$ not in D with $|\rho| = 4n$ are added to A. D contains all strings $< i, x, \rho >$ queried in earlier stages. Since there are less than $n \cdot 2^{n+1}$ earlier stages and on each stage at most 2^n strings are queried, $\#D < 2n \cdot 2^{2n}$.

(2) is obvious, since $< i, x, \rho >$ is added to A only if $\mathcal{M}_i^A$ accepts x within 2^n steps. □

Our next lemma shows that A is not only decidable in exponential time, but A does not add much power to deterministic time bounded machines. A universal set for all sets decidable in time 2^n with oracle A is itself decidable in exponential time. (From "$A \in \mathrm{DTIME}(2^n)$" we could only conclude "$\mathrm{DTIME}(2^n)^A \subseteq \mathrm{DTIME}(2^n)^{\mathrm{DTIME}(2^n)} = \mathrm{DTIME}(2^{2^n})$".)

Lemma 2. $L_A = \{\$^i x \mid \mathcal{M}_i^A \text{ accepts } x \text{ in time } 2^{|x|}, |x| > i\} \in \mathrm{DTIME}(2^{6n})$.

Proof. We construct a machine $\mathcal{M}$ which accepts L_A (without oracle) in time 2^{6n}.

On input $\$^i x$ $(i < |x|)$ $\mathcal{M}$ simulates all $\mathcal{M}_j^A(y)((j,y) \leq (i,x), j < |y|)$ for $2^{|y|}$ steps in the order of the oracle construction, recording the set D (as list of oracle strings) and the outcome of these machines. Oracle queries "$a \in A$?" are replaced by the following procedure:

(1) If a has not the form $< k, z, \rho >$ with $|z| > k$ and $|\rho| = 4|z|$, then $a \notin A$.

(2) Otherwise, if $a =< k, z, \rho >$ and $(k,z) \geq (j,y)$, then $a \notin A$. If $a \notin D$, $D := D \cup \{a\}$.

(3) Otherwise, if $a \in D$, then $a \notin A$.

(4) Otherwise $(a =< k, z, \rho >, (k,z) < (j,y), a \notin D)$, then $a \in A \iff \mathcal{M}_k^A$ accepts z (which was recorded on stage (k,z)).

If $\mathcal{M}_j^A(y)$ enters an accepting configuration, this fact is recorded and the next machine is simulated. After $2^{|y|}$ steps of simulation we know that $\mathcal{M}_j^A(y)$ does not accept within the time bound and record this fact. After stage (i,x) we know whether or not $\mathcal{M}_i^A$ accepts x within the time bound and thus have decided "is $\$^i x \in L_A$?".

Thus $\mathcal{M}$ accepts L_A.

On stage (i,x) D contains at most $2|x| \cdot 2^{2|x|}$ strings of length at most $2^{|x|}$. Thus the simulation of a single oracle query costs $O(|x| \cdot 2^{3|x|})$ steps. The other simulation steps are cheaper. Thus the simulation of $\mathcal{M}_j^A(y)$ $((j,y) \leq (i,x))$ can be done in $O(|x| \cdot 2^{4|x|})$ steps, which yields the total costs for all stages up to (i,x) of $O(|x|^2 \cdot 2^{5|x|}) \subseteq O(2^{6|x|})$. □

Proof of Theorem 1.

- "$\mathrm{EXTIME}^A = \mathrm{EXTIME}$"

  Suppose $L \in \mathrm{EXTIME}^A$. Then there is an oracle machine $\mathcal{M}_i^A$ and some k such that $\mathcal{M}_i^A$ decides L within $2^{k \cdot n}$ steps. Let $L' = \{x10^m \mid x \in L,\, m \geq k \cdot |x|\}$ be the appropriately padded set. Then $y = x10^m \in L'$ can be decided by some $\mathcal{M}_j^A$ in time $|y| + 2^m \leq 2^{|y|}$.

  Hence by Lemma 2,

$$\begin{aligned} L &= \{x \mid \$^j x 10^{k \cdot |x| + j} \in L_A\} = \{x \mid x10^{k \cdot |x| + j} \in L'\} \\ &\in \mathrm{DTIME}(2^{6(2j + (k+1) \cdot |x|)}) = \mathrm{DTIME}(2^{6(k+1) \cdot |x|}) \\ &\subseteq \mathrm{EXTIME}\,. \end{aligned}$$

- "$\mathrm{ZPTIME}(n)^A \supseteq \mathrm{EXTIME}^A$"

  Since EXTIME^A is closed under complement and $\mathrm{ZPTIME}(n)^A = \mathrm{RTIME}(n)^A \supseteq \mathrm{EXTIME}^A \cap \text{co-}\mathrm{RTIME}(n)^A$, it is sufficient to show "$\mathrm{RTIME}(n)^A \supseteq \mathrm{EXTIME}^A$".

  Suppose $L \in \mathrm{DTIME}(2^{k \cdot n})^A$.

  Let $L' = L(\mathcal{M}_j^A)$ as above, $\mathcal{M}_i^A$ runs in time 2^n. An R-machine $\mathcal{M}$ accepts $L = \{x \mid x10^{k \cdot |x| + j} \in L'\}$ as follows:

  On input x, $\mathcal{M}$ computes $y = x10^{k \cdot |x| + j}$. Then $\mathcal{M}$ chooses a random string ρ of length $4 \cdot |y|$. $\mathcal{M}$ accepts iff $< j, y, \rho > \in A$.

  Obviously, $\mathcal{M}$ runs in time $O(n)$. From Lemma 1 we conclude for all x:

$$\begin{aligned} x \notin L &\implies y \notin L' \implies \forall \rho\ < j, y, \rho > \notin A \implies \mathrm{Prob}[< j, y, \rho > \in A] = 0\,, \\ x \in L &\implies \mathrm{Prob}[< j, y, \rho > \in A] > \frac{3}{4}\,. \end{aligned}$$

- "$\mathrm{ZPTIME}(n)^A \subseteq \mathrm{EXTIME}^A$" is obvious:

$$\mathrm{ZPTIME}(n)^A \subseteq \mathrm{PrTIME}(n)^A \subseteq \mathrm{DTIME}(2^{O(n)})^A$$

  for any oracle A.

The corresponding statements for ZPP^A, $\mathrm{EXPTIME}^A$, and EXPTIME are proved in similar way using polynomial instead of linear padding. □

3 Oracle B with $\mathbf{BPP}^B = \Delta_2\mathbf{EXPTIME}^B = \Delta_2\mathbf{EXPTIME}$

The construction of the oracle B follows a similar idea as for the oracle A. The main difference is that we must introduce strings $< i, x, \rho >$ into the oracle before $\mathcal{M}_i^B(x)$ is encoded. This yields a small two-sided error for the probabilistic machine.

B will have the property that for all Δ_2-oracle machines $\mathcal{M}_i$ running in time 2^n and all x with $|x| = n > i$ the following holds:

$$\begin{aligned} x \in L(\mathcal{M}_i^B) &\implies \#\{\rho \in \Sigma^{4n} \mid < i, x, \rho > \in B\} > \frac{3}{4} \cdot 2^{4n} \\ x \notin L(\mathcal{M}_i^B) &\implies \#\{\rho \in \Sigma^{4n} \mid < i, x, \rho > \in B\} < \frac{1}{4} \cdot 2^{4n} . \end{aligned}$$

Furthermore

$$\Delta_2\mathrm{TIME}(2^n)^B \subseteq \Delta_2\mathrm{TIME}(2^{21n}) .$$

Again we can conclude

Theorem 2. *There exists an oracle B, such that*

$$\begin{aligned} BPTIME(n)^B &= \Delta_2 EXTIME^B = \Delta_2 EXTIME , \\ BPP^B &= \Delta_2 EXPTIME^B = \Delta_2 EXPTIME . \end{aligned}$$

Construction of the Oracle B

During the construction we record all oracle queries in (initially empty) sets B (strings with positive answer) and C (strings with negative answer), $B \cap C = \emptyset$. $E = \{0, 1, \$\}^* \setminus (B \cup C)$ contains all strings with yet undetermined outcome.

Recall that a Δ_2-machine with an oracle X is a deterministic machine which can query arbitrary nondeterministic linear time machines with the oracle X.

Let us denote the j-th nondeterministic linear time machine with oracle X by $\mathcal{N}_j^X$.

Stage $(0, \epsilon)$:

$B := \emptyset$;
$E := \{< i, x, \rho > \mid |\rho| = 4 \cdot |x|, i < |x|\}$;
$C := \{0, 1, \$\}^* \setminus E$;

Stage (i, x) $(i < |x|)$:

Simulate up to $2^{|x|}$ steps of $\mathcal{M}_i(x)$.

If $\mathcal{M}_i(x)$ queries "$y \in L(\mathcal{N}_j^B)$?", do the following:

If there is a set $D \subseteq E$ such that $y \in L(\mathcal{N}_j^{B \cup D})$, then $\mathcal{N}_j^{B \cup D}$ has at least one accepting path of length $|y|$. Suppose F is the set of all oracle queries on this path. Set $B := B \cup (F \cap D)$; $C := C \cup (F \cap (E \setminus D))$; $E := E \setminus F$. Otherwise for all $D \subseteq E$ $y \notin L(\mathcal{N}_j^{B \cup D})$.

If $\mathcal{M}_i$ accepts x, encode this:

$B := B \cup \{< i, x, \rho >\in E\}$; $E := E \setminus B$.

If $\mathcal{M}_i$ rejects x or does not accept x within $2^{|x|}$ steps

$C := C \cup \{< i, x, \rho >\in E\}$; $E := E \setminus C$. □

Lemma 3. *Suppose B is constructed as described above. Then for all i, x $(i < |x|)$ the following holds:*

(1) If $\mathcal{M}_i^B$ accepts $x \in \Sigma^n$ within 2^n steps, then $< i, x, \rho > \notin B$ for at most $2n \cdot 2^{3n}$ strings $< i, x, \rho >$ with $|\rho| = 4n$.

(2) If $\mathcal{M}_i^B$ does not accept $x \in \Sigma^n$ within 2^n steps, then $< i, x, \rho > \in B$ for at most $2n \cdot 2^{3n}$ strings $< i, x, \rho >$ with $|\rho| = 4n$.

Proof. At the end of stage (i, x) all $< i, x, \rho > \in E$ are added to B (case (1)) or to C (case (2)). Since there are less than $2n \cdot 2^n$ stages $(j, y) \leq (i, x)$, it is sufficient to show that at most 2^{2n} strings $< i, x, \rho >$ are removed from E during stage $(j, y) < (i, x)$ and during but before the end of stage (i, x).

$\mathcal{M}_j(y)$ $(j, |y| \leq n)$ performs at most 2^n queries of the form "$z \in L(\mathcal{N}_k^B)$?" with $|z| \leq 2^n$. For each of these queries the size of F (in the oracle construction) is bounded by $|z| \leq 2^n$. Thus for each query "$z \in L(\mathcal{N}_k^B)$?" at most $\#F \leq 2^n$ strings (possibly of the form $< i, x, \rho >$) are removed from E. At the end of stage (j, y) only strings $< j, y, \rho >$ are added to B or C and thus removed from E. □

Our next lemma asserts that B does not add much power to Δ_2-machines. The proof of this fact is much more difficult than the proof of the corresponding Lemma 2.

Recall that the construction of B does not completely determine the oracle B: when "$y \in L(\mathcal{N}_k^B)$" is fixed, we can choose different sets $D \subseteq E$ such that $y \in L(\mathcal{N}_j^{B \cup D})$ and for every such D select several accepting computation paths of $\mathcal{N}_j^{B \cup D}(y)$. Thus by appropriate choice arbitrary complex (even undecidable) oracle sets B may turn out. The proof of Lemma 4

yields the construction of one oracle set B consistent with the above described construction and thus with the properties of Lemma 3. In the rest of the paper B denotes this set.

Lemma 4. *There is an oracle B (which is one of the possible sets that turn out from the "construction of oracle B") such that*

$$L_B = \{\$^i x \mid \mathcal{M}_i^B \text{ is a } \Delta_2\text{-machine accepting } x \text{ in time } 2^{|x|},\ i < |x|\} \in \Delta_2 TIME(2^{21n}) .$$

Proof. Similar as in the proof of Lemma 2 the Δ_2-machine $\mathcal{M}$ with input $\$^i x$ will simulate all stages of the oracle construction up to stage (i, x).

Again we record the outcome of $\mathcal{M}_j^B(y)$ on stage (j, y) in a list Z. Some at the positive or negative answers to the oracle queries are recorderd in the additional lists X and Y. X and Y are (initially empty) lists of oracle strings of the form $< k, z, \rho >$ ($|\rho| = 4 \cdot |z|$, $k < |z|$) which are known to be in B (or not in B, respectively). Let E be the same set as in the "construction of oracle B", i.e. on stage (j, y)

$$E = \{< k, z, \rho > |\ (k, z) \geq (j, y),\ k < |z|, |\rho| = 4 \cdot |z|\} \setminus (X \cup Y) .$$

In order to simulate queries "$z \in L(\mathcal{N}_k^B)$?" we will use the following universal set L, which determines on stage (j, y) whether or not there is an augmentation of the current B consistent with the previous oracle queries (recorded in X, Y, Z) such that the nondeterministic machine $\mathcal{N}_k^{B \cup D}$ starting in same configuration C can reach an accepting configuration within t steps:

$\$^k c\$X\$Y\$Z\$^j\$y\$^t \in L \Longleftrightarrow$ $\mathcal{N}_k$ starting in configuration accepts within t steps, where the oracle queries "$a \in B$?" are replaced as follows:

(1) "$a \notin B$" if a has not the form $< l, u, \rho >$ with $l < |u|$, $|\rho| = 4 \cdot |u|$. For the other cases assume $a =< l, u, \rho >$.

(2) "$a \in B$" if a is contained in the list X or if $(l, u) < (j, y)$, $a \notin Y$ and $(l, u) \in Z$.

(3) "$a \notin B$" if $a \in Y$ or if $(l, u) < (j, y)$, $a \notin X$ and $(l, u) \notin Z$.

(4) Otherwise ($a \in E$) replace the query by a nondeterministic choice.

It is easy to see that $L \in \text{NTIME}(k \cdot t \cdot (|X| + |Y| + |Z|)) \subseteq \text{NTIME}(n^3)$.

Suppose we are simulating stage $(j, y) \leq (i, x)$.

Using L a query of $\mathcal{M}_j(y)$ of the form "$z \in L(\mathcal{N}_k^B)$?" can be replaced by a sequence of queries "$p \in L$?", which yields a stepwise simulation of an accepting path of N_k^B, and appropriate augmentation of X or Y:

$c :=$ initial configuration of $\mathcal{N}_k(z)$;
$t := |z|$;
IF $\$^k c\$X\$Y\$Z\$^j\$y\$^t \in L$ THEN
WHILE $t > 0$ AND c is not accepting DO
BEGIN
  IF the next step of $\mathcal{N}_k$ is an oracle query "$a \in B$?" and a is not yet recorded THEN
  BEGIN
    add a to X;
    $c' :=$ next configuration if $a \in B$;
    IF $\$^k c'\$X\$Y\$Z\$\$^j Y\$^{t-1} \notin L$ THEN
    BEGIN
      remove a from X;
      add a to Y;
      $c' :=$ next configuration if $a \notin B$
    END
  END
  ELSE { the next step is a nondeterministic choice }
    determine a next configuration c' with $\$^k c'\$X\$Y\$Z\$\$^j y\$^{t-1} \in L$
    { at least one c' has this property };
  $t := t - 1$;
  $c := c'$
END;
IF c is an accepting configuration of $\mathcal{N}_k$
THEN "$z \in L(\mathcal{N}_k^B)$" ELSE "$z \notin L(\mathcal{N}_k^B)$".

At the end of the stage (j, y) (i.e. when $\mathcal{M}_j$ reaches an accepting configuration or else after $2^{|y|}$ simulation steps) record the outcome of $\mathcal{M}_j(y)$: if $\mathcal{M}_j$ accepts y within $2^{|y|}$ steps, add (j, y) to Z.

The oracle B constructed by this procedure is determined by

$$a \in B \iff a =< i, x, \rho >,\ i < |x|,\ |\rho| = 4 \cdot |x| \text{ and after stage } (i,x) \text{ of the simulation}$$
$$\text{either } a \in X \text{ or } a \notin Y \text{ and } (i,x) \in Z.$$

On stage $(j, y) \leq (i, x)$ $\mathcal{M}$ simulates at most 2^n ($n = |x|$) steps. The simulation of a query "$z \in L(\mathcal{N}_k^B)$" costs $|z| \leq 2^n$ steps and 2^n queries to the oracle B times the cost for looking at and updating the lists X, Y and Z. These can contain up to $2n \cdot 2^{3n}$ elements of length 2^{2n}. Thus the total costs for all $2n \cdot 2^n$ stages up to (i, x) are bounded by $4n^2 \cdot 2^{6n}$ and

$$L_B \in \mathrm{DTIME}(2^{7n})^{\mathrm{NTIME}(n^3)} \subseteq \mathrm{DTIME}(2^{21n})^{\mathrm{NTIME}(n)} = \Delta_2\mathrm{TIME}(2^{21n}) .$$

□

Proof of Theorem 2. Follows from Lemma 3 and Lemma 4 in the same way as Theorem 1 from Lemma 1 and Lemma 2. □

4 Consequences

The sets A and B have many interesting properties. Perhaps the most interesting is that randomized reduction can be exponentially more powerful than deterministic or nondeterministic reduction (cf. [AM 77]).

Definition (Reducibilities)

$X \leq_\gamma Y :\Leftrightarrow$ there is a polynomial time bounded NTM $\mathcal{M}$ with:

(1) For every input x there is at least one computation which produces an output.

(2) $\forall (x,y)\, \mathcal{M}(x) = y \Rightarrow [x \in X \Leftrightarrow y \in Y]$

$X \leq_{UR} Y :\Leftrightarrow$ there is a polynomial time bounded PTM $\mathcal{M}$ with:

(1) every computation produces an output

(2a) $x \in X \Rightarrow \mathcal{M}(x) \in Y$

(2b) $x \notin X \Rightarrow \mathrm{Prob}[\mathcal{M}(x) \notin Y] > \frac{3}{4}$ (unfaithful R-reduction)

$X \leq_{BPP} Y :\Leftrightarrow$ as $\leq_{UR}$ with (1'), (2b), and (2a')

(2a') $x \in X \Rightarrow \mathrm{Prob}[x \in X \Rightarrow \mathcal{M}(x) \in Y] > \frac{3}{4}$.

Obviously $X \leq_\gamma Y \Rightarrow X \in NP^Y$, $X \leq_{\mathrm{DTIME}(T)} Y \Rightarrow X \in \mathrm{DTIME}(T)^Y$.

Theorem 3. *UR-Reductions are exponentially more powerful than DTIME-reductions.*

(1) $\forall X \in EXPTIME,\ X \leq_{UR} \bar{A}$ *and* $\bar{X} \leq_{UR} \bar{A}$

(2) $\forall k \forall T \in O(2^{n^k})\ \exists X \in EXPTIME,\ X \not\leq_{DTIME(T)} \bar{A}$.

Proof.

(1) follows (by polynomial padding) from Lemma 1.

(2) Suppose $T \in O(2^{n^k})$.
$X \leq_{\mathrm{DTIME}(T)} \bar{A} \Rightarrow X \in \mathrm{DTIME}(T)^{\bar{A}} \Rightarrow X \in \mathrm{DTIME}(2^{6 \cdot n^k})$ (by Lemma 2), which is not true for all $X \in \mathrm{EXPTIME}$. □

Theorem 4. *BPP-reductions are exponentially more powerful than nondeterministic (and than γ-) reductions (cf. [AM 77]):*

(1) $\forall X \in \Delta_2 EXPTIME,\ X \leq_{BPP} B$

(2) $\forall k, \forall T \in O(2^{n^k}),\ \Delta_2 TIME(T)^B \not\supseteq \Delta_2 EXPTIME.$

Proof.

(1) follows from Lemma 3.

(2) as Proof of Theorem 3 using Lemma 4. □

Since the oracle queries used for the $\leq_{UR}$ and $\leq_{BPP}$ are extremely simple, they can be computed by NC^1-circuits. Thus $\leq_{UR}$ and $\leq_{BPP}$ in Theorems 3 and 4 can be replaced even by Las Vegas NC^1-reducibility and Monte Carlo NC^1-reducibility (defined in an obvious way).

Since Monte Carlo machines have small nonuniform circuits it is an easy consequence from Lemma 3 that relative to oracle B, Δ_2EXTIME has linear size circuits (for the weaker version of it see [W 83], cf. also [K 82]).

We list some other consequences for our oracles with hints how to prove them (to shorten the formulas we denote EXPTIME by E):

(1) $P^A \subsetneq \mathrm{ZPP}^A = N = PH^A = E = E^A \subsetneq ZPE^A = E^E = \mathrm{DTIME}(2^{2^n})$.

(2) $P^B \subsetneq NP^B \cap coNP^B \subsetneq NP^B \subsetneq \Delta_2 P^B \subsetneq \mathrm{BPP}^B = \Sigma_2 P^B = PH^B = \Delta_2 E = \Delta_2 E^B \subsetneq BPE^B = \Sigma_2 E^B = EH^B = E^{\Delta_2 E} = \Delta_2\mathrm{TIME}(2^{2^n})$.

(The inclusions 1 to 4 are strict because otherwise the polynomial hierarchy collapses and $\Delta_2 P^B = \Sigma_2 P^B = \Delta_2 E^B$, which is impossible.)

(3) If $E \neq ZPE$, then $P^A \not\supseteq \mathrm{ZPP}$.
($P^A \supseteq \mathrm{ZPP} \Rightarrow E = E^A = E^{P^A} \supseteq E^{\mathrm{ZPP}} = ZPE$)

(4) If $\Delta_2 E \neq \Delta_3 E$, then $P^B \not\supseteq NP$ (i.e. B is not NP-hard, but complete for $\Delta_2 E$ under $\leq_{\mathrm{BPP}}$). ($P^B \supseteq NP \Rightarrow \Delta_2 E = \Delta_2 E^{P^B} \supseteq \Delta_2 E^{NP} = \Delta_3 E$.)

(5) If $\Delta_2 E \neq \Delta_3 E$, then $NP^B \not\supseteq coNP$.
($NP^B \supseteq coNP \Rightarrow NP^{NP} \subseteq NP^B \Rightarrow \Delta_3 E \subseteq \Delta_2 E^B = \Delta_2 E$)

(6) If $\Delta_2 E \neq \Sigma_2 E$, then $\Delta_2 P^B \not\supseteq \Sigma_2 P$.
($\Delta_2 P^B \supseteq \Sigma_2 P \Rightarrow \Delta_2 E = \Delta_2 E^B = E^{\Delta_2 P^B} \supseteq E^{\Sigma_2 P} = \Sigma_2 E$)

5 Conclusion

Our results show that the randomized computation can be extremely powerful when compared with deterministic computation in a relativized context, even though randomization has almost no additional power in the presence of random oracles.

We have constructed oracles A and B with maximal collapse between polynomial and exponential classes without known strict inclusion:

$$\begin{aligned} \mathrm{ZPP}^A &= \Sigma_1 P^A \cap \Pi_1 P^A = \Delta_1 E^A = \Delta_1 E \ (= \mathrm{EXPTIME}), \\ \mathrm{BPP}^B &= \Sigma_2 P^B \cap \Pi_2 P^B = \Delta_2 E^B = \Delta_2 E . \\ & \quad (\mathrm{BPP}^B \subseteq \Sigma_2 P^B \cap \Pi_2 P^B, \text{ see [S 83]}) \end{aligned}$$

It is an open question, if such oracles with maximal collapse exist also on other levels of the polynomial and exponential hierarchies, i.e. whether there exists C such that for some $k > 2$,

$$\Sigma_k P^C \cap \Pi_k P^C = \Delta_k E^C = \Delta_k E .$$

It seems that the methods presented in this paper cannot be applied directly to higher levels, since no probabilistic class is known below $\Sigma_k P$ and not below $\Delta_k P$ $(k > 2)$.

Acknowledgments. We thank Eric Allender, Klaus Ambos-Spies, Richard Beigel, Yuri Gurevich, and Johan Håstad for the number of interesting discussions connected to the topic of this paper.

References

[A 78] Adleman, L., *Two Theorems on Random Polynomial Time*, Proc. 19^{th} IEEE FOCS, 1978, pp. 75-83.

[AM 77] Adleman, L., Manders, K., *Reducibility, Randomness, and Intractibility*, Proc. 9^{th} ACM STOC, 1977, pp. 151-163.

[ABHH 92] Allender, E., Beigel, R., Hertrampf, U., Homer, S., *Almost-Everywhere Complexity Hierarchies for Nondeterministic Time*, Manuscript, 1992; A preliminary version has appeared in Proc. STACS '90, LNCS 415, Springer-Verlag, 1990, pp. 1-11.

[BG 81] Bennett, Ch. H., Gill, J., *Relative to a Random Oracle A, $P^A \neq NP^A \neq co-NP^A$ with Probability* 1, SIAM J. on Computing **10**, 1981, pp. 96-113.

[F 92] Fortnow, L., *Personal Communication*, 1992.

[FS 89] Fortnow, L., Sipser, M., *Probabilistic Computation and Linear Time*, Proc. 21^{st} ACM STOC, 1989, pp. 148-166.

[F 79] Freivalds, R., *Fast Probabilistic Algorithms*, Proc. MFCS'79, LNCS **75**, 1979, Springer-Verlag, pp. 57-69.

[Jo 90] Johnson, P.S., *A Catalog of Complexity Classes*, in Handbook of Theoretical Computer Science, Vol. A., Algorithms and Complexity, Elsevier-MIT Press, 1990, pp. 69-161.

[K 82] Kannan, R., *Circuit-Size Lower Bounds and Non-reducibility to Sparse Sets*, Information and Control **55**, 1982, pp. 40-46.

[KV 87] Karpinski, M., Verbeek, R., *On the Monte Carlo Space Constructible Functions and Separation Results for Probabilistic Complexity Classes*, Information and Computation **75**, 1987, pp. 178-189.

[KV 88] Karpinski, M., Verbeek, R., *Randomness, Provability, and the Separation of Monte Carlo Time and Space*, LNCS 270, Springer-Verlag, 1988, pp. 189-207.

[R 82] Rackoff, C., *Relational Questions Involving Probabilistic Algorithms*, J. ACM **29**, 1982, pp. 261-268.

[S 83] Sipser, M., *A Complexity Theoretic Approach to Randomness*, Proc. 15^{th} ACM STOC, 1983, pp. 330-335.

[St 85] Stockmeyer, L., *On Approximation Algorithms for $\#P$*, SIAM J. Comput. **14**, 1985, pp. 849-861.

[W 83] Wilson, C., *Relativized Circuit Complexity*, 24^{th} IEEE FOCS, 1983, pp. 329-334.

[Y 90] Yao, A. C., *Coherent Functions and Program Checkers*, Proc. 22^{nd} ACM STOC, 1990, pp. 84-94.