JOURNAL OF COMPUTER AND SYSTEM SCIENCES 22, 1–8 (1981)

Some Results on Relativized Deterministic and Nondeterministic Time Hierarchies

SHLOMO MORAN

Department of Computer Science,
University of Minnesota, Minneapolis, Minnesota 55455

Received March 18, 1980

Relativized forms of deterministic and nondeterministic time complexity classes are considered. It is shown, at variance with recently published observations, that one basic step by step simulation technique in complexity theory does not relativize (i.e., does not generalize to computations with oracles), using the current definitions for time and space complexity of oracle computations. A new natural definition is given with respect to which this simulation does relativize. Further results in the paper are obtained with respect to this new definition. It is shown that for each "well-behaved" class of functions, τ, there is a set E such that for each function $t(n) \in \tau$, there is a set which is accepted in $t(n)$ time by a nondeterministic Turing machine with oracle E, but is not accepted in less than $O(2^{t(n)})$ time by any deterministic Turing machine with oracle E, and there is another set which is accepted in $t(n)$ time by a deterministic Turing machine with oracle E, but is not accepted in less than $O(t(n))$ time by any nondeterministic Turing machine with oracle E. The techniques used are based on refinements of diagonalization techniques of Baker *et al.* (*SICOMP* (1975), 421–442) and Ladner and Lynch (*Math. Systems Theory* **10** (1976), 19–32). One immediate corollary is a generalization of a theorem of Baker *et al.* (*SICOMP* (1975), 421–442), which asserts that for some set E, $P^E \neq NP^E$. Other applications concerning deterministic and nondeterministic relativized time hierarchies are given.

1. INTRODUCTION

The problem whether the class of sets which can be recognized by deterministic Turing machines in polynomial time is equal to the class of sets which can be recognized by non-deterministic Turing machines in polynomial time (generally referred to as the "$P = ?NP$" problem), has long been known for its hardness [3, 6, 1, 5]. Some researchers tend to explain our difficulty in solving this problem by the following observations (originating in [2]):

OBSERVATION (1a). *The $P = ?NP$ problem may have different anwswers under different relativizations (i.e., for some sets A and B, $P^A = NP^A$ and $P^B \neq NP^B$. $P^E(NP^E)$ denote the class of sets accepted in polynomial time by (non)deterministic Turing machines with oracle set E).*

OBSERVATION (1b). *The methods known today of proving or disproving equality between classes of sets relativize (i.e., proving by those methods that $P = NP$ ($P \neq NP$) would imply that $P^E = NP^E$ ($P^E \neq NP^E$) for arbitrary set E).*

0022-0000/81/010001–08$02.00/0

Thus, proving that $P = NP$ ($P \neq NP$) by a method satisfying observation (1b) will contradict observation (1a).

A similar argument has been used to provide an evidence for the difficulty of other hard problems in complexity theory, e.g., the following problem:

Assuming that $P \neq NP$, is the Class R, of randomly recognizable in polynomial time sets, equal to P? (See [9].)

The validity of observation (1b) above may depend on the exact definition of time (or space) complexity of oracle computations: In [7] a definition of space complexity is given, under which certain theorems, involving Nlog space complexity classes, do not relativize. In this paper it is shown that under the standard definition of the complexity of oracle computations, as used in [2], the linear speed-up theorem does not relativize. We give a broader definition for time and space complexity of oracle computations, under which all the above theorems relativize, and this definition is used in the rest of the paper. (Our results, however, concern only time complexity classes.)

Failing to solve the $P = ?NP$ problem, it still could be hoped that some nontrivial upper bound on the time complexity of NP could be found (e.g., that each problem in NP could be solved in $O(2^n)$ time—see [1, p. 403]). In this paper we show that, under certain relativization, no such nontrivial upper bound exists. This implies that proving such a nontrivial upper bound, if at all possible, requires a technique which does not relativize, and hence it is probably not substantially easier than solving the $P = ?NP$ problem. We also show that proving that nondeterminism is always faster than determinism (i.e., that each set which is recognizable by a deterministic Turing machine in $t(n)$ time, where $t(n) > O(n)$,[1] can be recognized in less than $O(t(n))$ time by a nondeterministic Turing machine) requires a technique which does not relativize. Some applications on the deterministic and nondeterministic time hierarchies under relativizations are also given.

In Section 2, the preliminary definitions of this paper are given. In Section 3, Lemma 2, we show that, when using the standard definition for time complexity of computations with oracle set E, the linear speed-up theorem does not relativize. The diagonalization technique used to obtain the results of his paper is introduced in this lemma. We then give an alternative, broader definition for time (and space) complexity of oracle computations, and the results of this paper apply to this broader definition. In section 4, the main theorems of this paper are given, together with several applications.

2. Preliminary Definitions

Let Σ be a finite alphabet. Σ^* is the set of all finite strings over Σ. All sets mentioned in the sequel are subsets of Σ^*, for some Σ. For a string $x \in \Sigma^*$, $l(x)$ denotes the length of x.

[1] For functions $t(n)$, $s(n)$, $t(n) > O(s(n))$ denotes $\lim_{n\to\infty}(s(n)/t(n)) = 0$.

Our models of computations are multi tape, deterministic and nondeterministic, Turing machines acceptors, which have one semi-infinite read-only input tape and a finite number of semi-infinite work tapes. We shall denote (non)deterministic Turing machines by (N)DTM. The set accepted by an (N)DTM M is the set $\{x \mid$ there is a computation of M on input x which terminate when M is in an accepting state$\}$. We shall identify Turing machines with the sets accepted by them. For a more detailed exposition of Turing machines the reader is referred to [5].

An oracle Turing machine is a Turing machine which has a distinguished worktape, called the "oracle tape," and three distinguished states—$q_?$, q_{yes} and q_{no}. The computations of oracle Turing machine are carried with respect to some oracle set E. When the machine is at state $q_?$, it enters state q_{yes} (q_{no}) if the string written on the oracle tape at that time is in E (not in E). All other transitions rules of the machine are the same as that of a regular Turing machine. For an oracle Turing machine M and a Set E, M^E denotes the set accepted by M with oracle Set E. Throughout the paper, $(T_1, T_2, \ldots)$ is an enumeration of the oracle DTMs, and $(NT_1, NT_2, \ldots)$ is an enumeration of the oracle NDTMs.

For a function $t: Z^+ \to R^+$ (Z^+ denotes the nonnegative integers, R^+ denotes the nonnegative real numbers), $D(t(n))^E$ $[ND(t(n))^E]$ denotes the class of sets accepted by oracle DTMs (NDTMs) with oracle set E, which always stops within $t(n)$ steps on all inputs of length n. $\text{DSPACE}(t(n))^E$ ($\text{NSPACE}(t(n))^E$) denotes the class of sets accepted by oracle DTMs (NDTMs) with oracle Set E, which for all inputs of length n, always halts with no more than $t(n)$ cells being visited on any work tape (including the orcale tape.)

A function $t(n) \geqslant n$ is fully time constructible if there exists a DTM which for all n halts in exactly $t(n)$ steps on all inputs of length n (or, equivalently, if there exists a DTM which for all n, halts in exactly $t(n)$ steps on input 0^n). A set τ of fully time constructible functions is recursively enumerable and $(t_1(n), t_2(n), \ldots)$ is an enumeration of τ if:

(i) For each $t(n) \in \tau$, $t(n) = t_i(n)$ for some i, and

(ii) There is an enumeration $(M_1, M_2, \ldots)$ of DTMs. s.t. for each i and n, $t_i(n) =$ the number of steps that M_i make before halting on input O^n.[2]

The set of polynomial bounded, fully time constructible functions is recursively enumerable, as can be shown by the following enumeration $(t_i(n))$, where

$$
\begin{aligned}
t_i(n) &= n && \text{if } T_i \text{ on input } 0^n \text{ halts in less than } n \text{ steps} \\
&= i + n^i && \text{if } T_i \text{ on input } 0^n \text{ does not halt within } i + n^i \text{ steps} \\
&= \phi_i(0^n) && \text{otherwise,}
\end{aligned}
$$

where T_i is the ith DTM, and $\phi_i(x)$ is the number of steps T_i makes before halting on input x. ($\phi_i(x) = \infty$ if T_i diverges on x.)

[2] It is not hard to show that the class of all fully time constructible functions is not recursively enumerable.

For a Turing machine M and a fully time constructible function $t(n)$, "M with a clock $t(n)$" is a machine which on each input of length n simulates M up to $t(n)$ steps, and then halts.

3. Oracle Computations and the Linear Speed-Up Theorem

The linear speed-up theorem," due to Hartmanis and Stearn [4], is the following:

Lemma 1. *Let $t(n)$ be a function s.t. $t(n) > O(n)$. Then for each positive constant C, $D(t(n)) = D(Ct(n))$ $[ND(t(n)) = ND(Ct(n))]$.*

Proof. See [5, Theorem 12.3]. ∎

In contradiction to observation (1b) in Section 1, Lemma 1 above does not relativize, as follows from the following surprising result:

Lemma 2. *Let τ be a recursively enumerable class of fully time constructible functions. Then there exists a set E s.t. for each $t(n) \in \tau$, $D(t(n)+2)^E \not\subseteq ND(t(n))^E$.*

Proof. The set $E \subset \{0,1\}^*$ is constructed in stages, where at stage m at most one string is inserted into E. Let $E(m)$ denote the set of strings inserted into E before stage m. $E(0) = \phi$. Let $(t_1, t_2, \ldots)$ be an enumeration of τ, and let $I = ((i_1,j_1)(i_2,j_2),\ldots)$ be an enumeration of the pairs of nonnegative integers.

For each pair (i,j), NT_{ij} denotes the ith nondeterministic Turing machine, to which a clock of time $t_j(n)$ is attached. Finally, let $e(m)$ be a fast growing function s.t. $e(0) = 1$ and for $m > 1$ $t_{j_m}(e(m)) > t_{j_k}(e(k))$ for all $k < m$. ((i_k,j_k) is the kth pair in I.) Such a function exists since for each k, $t_{j_k}(n) \geqslant n$ for all n.

Stage m is as follows: Run $NT_{i_m j_m}^{E(m)}$ on $0^{e(m)}$. If this accepts, then do nothing, else add $1^{t_{j_m}(e(m))}$ to E. (Note that at stage k, for $k \leqslant m$, no string of length $t_{j_m}(e(m))$ is queried about, since $t_{j_k}(e(k)) < t_{j_m}(e(m))$ for $k < m$, and within the $t_{j_m}(e(m))$ computation steps at stage m, only strings of length $< t_{j_m}(e(m))$ are queried about.)

For a given function $t(n) \in \tau$, a set $E_{t(n)}$ is defined by

$$E_{t(n)} = \{x \mid 1^{t(l(x))} \in E\}. \tag{2.1}$$

$E_{t(n)} \in D(t(n)+2)^E$, since $E_{t(n)}$ is recognized by a Turing machine which on input x writes $1^{t(l(x))}$ on the oracle tape, which can be done in $t(l(x))$ steps due to the fully time constructibility of $t(n)$, and then queries about $1^{t(l(x))}$, and accepts iff $1^{t(l(x))} \in E$ (i.e., q_{ycs} is the accepting state.) That can be done in another two steps.

Suppose now that $E_{t(n)}$ is accepted by the ith nondeterministic Turing machine NT_i in $t(n)$ time, and let j be such that $t_j(n)) = t(n)$. Let (i,j) be the mth pair in I. Then $0^{e(m)} \in E_{t(n)} \leftrightarrow 0^{e(m)} \in NT_{ij}^{E(m)} \leftrightarrow 0^{e(m)} \in NT_{ij}^{E} \leftrightarrow 1^{t(e(m))} \notin E \leftrightarrow 0^{e(m)} \notin E_{t(n)}$, a contradiction. This complete the proof of the lemma. ∎

Since clearly $D(t(n))^E \subseteq ND(t(n))^E$, Lemma 2 above implies that $D(t(n)+2)^E \not\subseteq D(t(n))^E$, and hence $D(t(n))^E \subsetneq D(Ct(n))^E$ for each constant $C > 1$—which means that Lemma 1 (the linear speed-up theorem) does not relativize.

The proof of the linear speed up theorem is based on the following observation: In order to predict the next m moves of a Turing machine T, the only information needed is the current state, and the content of $2m-1$ cells in each tape—the cell currently scanned by the head, and $m-1$ cells to the right and to the left of this cell. Hence, by encoding m cells into one, a fast simulation is possible. The above observation does not hold for oracle computations, since each query step depend on the content of the whole nonblank portion of the oracle tape. Therefore, it seems that just encoding of m cells of the oracle tape into one does not guarantee a speed up of the computation. To overcome this deficiency, a model of oracle computation which "attribute" the speed up property to the oracle tape is needed. Such a model is given in the following:

DEFINITION 3.1. Oracle Turing machine with oracle tape compression is an oracle Turing machine which, for some constant C, can make up to C moves of the oracle head in one time unit, and can compress up to C cells of the oracle tape into one space unit.

For an oracle set E and a function $t(n)$, $D(t(n))^E_{\text{com}}$ denotes the class of sets accepted by a deterministic oracle Turing machine with oracle tape compressing, using oracle set E, in time $t(n)$. $ND(t(n))^E_{\text{com}}$, $\text{DSPACE}(t(n))^E_{\text{com}}$ and $\text{NDSPACE}(t(n))^E_{\text{com}}$ are defined similarly.

LEMMA 1′. *For any function $t(n)$ s.t. $t(n) > O(n)$, for any oracle set E and for any positive constant C, $D(t(n))^E_{\text{com}} = D(Ct(n))^E_{\text{com}} \cdot (ND(t(n))^E_{\text{com}} = ND(Ct(n))^E_{\text{com}})$.*

Proof. By a straight forward generalization of the technique used to prove Lemma 1 ([5, Theorem 12.3]) to computations with oracle with tape compression. ∎

Some results resembling Lemma 2 appear in [7]. For instance, it is shown that for some Set E, Nlogspace^E (the sets accepted by nondeterministic Turing machine with oracle E in logarithmic space) is not included in P^E, while it is known that $\text{Nlogspace} \subseteq P$. In the model of oracle computation used in that paper, a distinction is made between the oracle tape and the work tapes: in a space bounded computation, the work space of the oracle tape is not counted. In other models of oracle computations, where the space (and time) complexity of the oracle tape are counted, the "$\text{Nlogspace} \subseteq P$" theorem, as well as other results which concern Nlogspace complexity which appear in [7], do relativize.

It seems that the model of oracle computations with tape compression introduced above, is the one which fits best to observation (1b), Section 1. The results of this paper are, therefore, proved with respect to this model.

The following converse of Lemma 1′ will be used later:

LEMMA 3. *If* $A \in D(t(n))^E_{\text{com}}$ $[A \in ND(t(n))^E_{\text{com}}]$, *then for some constant* C, $A \in D(Ct(n))^E$ $[A \in ND(Ct(n))^E]$.

Proof. Omitted. ∎

4. MAIN RESULTS

All our results concern oracle computations with tape compression, as defined in Definition 3.1, and are proved in two stages: First we prove a lemma concerning the standard definition of time complexity of oracle computation, as given in Section 2, and then we use Lemma 3 above to obtain a result concerning that broader definition. Thus, taking Lemma 2 to be the first stage, and Lemma 3 the second, we can prove:

THEOREM 1. *Let* τ *be a recursively enumerable class of fully time constructible functions. Then there is a set* E, *s.t. for each function* $t(n) \in \tau$ *and for each function* $s(n)$ *s.t.* $t(n) > 0(s(n))$, *it holds true that* $D(t(n))^E_{\text{com}} \not\subseteq ND(s(n))^E_{\text{com}}$.

Proof. Let E be the set contructed in Lemma 2. For $t(n) \in \tau$, let $E_{t(n)}$ be the set defined in (2.1). By Lemma 2, $E_{t(n)} \in D(t(n)+2)^E$, and hence, by Lemma 1′, $E_{t(n)} \in D(t(n))^E_{\text{com}}$. If $D(t(n))^E_{\text{com}} \subseteq ND(s(n))^E_{\text{com}}$ for some $s(n)$ s.t. $t(n) > O(s(n))$, then $E_{t(n)} \in ND(s(n))^E_{\text{com}}$, this implies, by Lemma 3, that $E_{t(n)}$ is in $ND(Cs(n))^E$ for some constant C. Since $t(n) > O(s(n))$, $t(n) > Cs(n)$ for all but a finite number of n's. Hence $E_{t(n)}$ is in $ND(t(n))^E$, in contradiction to Lemma 2. ∎

Theorem 1 above implies that under certain relativizations, there exist sets which cannot be recognized by nondeterministic algorithms faster than by deterministic ones. We now show that for some sets, under other relativizations, it is almost the other way around: It is known that if a set A is recognizable in $t(n)$ time by an NDTM, then for some constant C, A is recognizable in $2^{Ct(n)}$ time by some DTM ([5, Theorem 12.10]). This result easily relativizes to computations with oracles. The next lemma shows that under some relativization, no better upper bound on the deterministic time complexity of nondeterministic computations does exist.

LEMMA 4. *Let* τ *be a recursively enumerable class of fully time constructible functions. Then there is a set* E *s.t. for each* $t(n) \in \tau$, $ND(t(n)+2)^E \not\subseteq D(2^{t(n)})^E$.

Proof. The technique of the proof is similar to that of Lemma 2, and is sketched below:

Let $(t_1, t_2, \ldots)$ be an enumeration of τ, and let $I = ((i_1, j_1), (i_1, j_2) \cdots)$ be an enumeration of the pairs of nonnegative integers. For each pair (i, j) let T_{ij} be the ith deterministic Turing machine to which a clock of time $2^{t_j(n)}$ is attached. Let $e(m)$ be a function satisfying $e(m) > 2^{t_{j_k}(e(k))}$ for all $k < m$, where (i_k, j_k) is the kth pair in I, and let $E(m)$ be the set of strings inserted into E before stage m. Stage m is as follows:

Run $T_{i_m j_m}$ on $0^{e(m)}$. If this accepts then do nothing, else add one string of length $t_{j_m}(e(m))$ which was not queried about to E. Such a string exists since in 2^n steps an oracle Turing machine queries about less than 2^n strings, all of them are of length $<2^n$.

Now for a given fully time constructable function $t(n)$, define a set $E_{t(n)} = \{x \mid \exists y \in E, l(y) = t(l(x))\}$. By an argument similar to the one appearing in Lemma 2, it can be shown that $E_{t(n)} \in ND(t(n)+2)^E$, and $E_{t(n)} \notin D(2^{t(n)})_E$. ∎

Using Lemmas 3 and 4, the following theorem can be proved similarly to Theorem 1:

THEOREM 2. *Let τ be a recursively enumerable class of fully time constructible functions. Then there exists a Set E s.t. for each function $t(n) \in \tau$, and for each function $s(n)$ s.t. $2^{t(n)} > O(s(n))$, $ND(t(n))^E_{\text{com}} \not\subset D(s(n))^E_{\text{com}}$.* ∎

Combining the diagonalization techniques of Lemmas 2 and 4 (e.g., by executing stage m of Lemma 2 at stage $j = 2m$, and stage m of Lemma 4 at stage $j = 2m+1$), it is possible to obtain a set E which satisfies both Theorem 1 and Theorem 2, which results in:

THEOREM 3. *Let τ be a recursively enumerable class of fully time constructible funtions. Then there is a set E s.t. for each $t(n) \in \tau$:*

(a) $ND(t(n))^E_{\text{com}} \not\subseteq D(s(n))^E_{\text{com}}$ *for each* $s(n)$ *s.t.* $2^{t(n)} > O(s(n))$.

(b) $D(t(n))^E_{\text{com}} \not\subseteq ND(s(n))^E_{\text{com}}$ *for each* $s(n)$ *s.t.* $t(n) > O(s(n))$.

Proof. Omitted. ∎

An interesting corollary of Theorem 3 above concerns the relativization of the deterministic and nondeterministic time hierarchies:

COROLLARIES 1. *Let τ and E be as in Theorem 3. Then for each $t(n) \in \tau$ and for each $s(n)$ s.t. $t(n) > O(s(n))$, it holds true that $D(s(n))^E_{\text{com}} \subsetneq D(t(n))^E_{\text{com}}$ and $ND(s(n))^E_{\text{com}} \subsetneq ND(t(n))^E_{\text{com}}$.*

Proof. Clearly $D(s(n))^E_{\text{com}} \subseteq D(t(n))^E_{\text{com}}$ and $ND(s(n))^E_{\text{com}} \subseteq ND(t(n))^E_{\text{com}}$. On the other hand, by Theorem 3, $D(t(n))^E_{\text{com}} \not\subseteq ND(s(n))^E_{\text{com}}$, which implies that $D(t(n))^E_{\text{com}} \not\subseteq D(s(n))^E_{\text{com}}$, and $ND(t(n))^E_{\text{com}} \not\subseteq ND(s(n))^E_{\text{com}}$. ∎

Due to the linear speed-up theorem, the result above is, in a sense, the finest possible partition of the deterministic and nondeterministic time hierarchies. (For the unrelativized case, result almost as strong was obtained for the nondeterministic hierarchy in [10].) This result fits with our intuition that perhaps there are no gaps in the time hierarchy for "natural" complexity functions.

5. Conclusion

We have shown that under certain relativizations, there is no "uniform" relation between deterministic and nondeterministic computations: Some sets are accepted by NDTMs much faster than by DTMs, and for other sets, nondeterminism does not speed up their deterministic accepting time. The model of time complexity of oracle computation used is very close to the model of time complexity without oracles, which implies that proving a result for computations without oracles which contradict the above result is probably not easy, if at all possible.

The most interesting problem related to this paper is, perhaps, the following: is there a result concerning the relations between time (or space) complexity classes, which do not relativize to the model of oracle computations with tape compression? Such a result is necessary for solving the $P = ?NP$ problem, and other problems mentioned in the text.

References

1. A. V. Aho, J. E. Hopcroft, and J. D. Ullman, "The Design and Analysis of Computer Algorithms," Addison–Wesley, Reading, Mass., 1974.
2. T. Baker, J. Gill, and R. Solovay, Relativization of the $P = ?NP$ problem, *SICOMP* (1975), 421–442.
3. S. A. Cook, The complexity of theorem proving procedures, *in* "Proceedings, 3rd STOC," pp. 151–158.
4. J. Hartmanis and R. E. Stearns, On the computational complexity of algorithms, *Trans. Amer. Math. Soc.* **117** (1965), 285–306.
5. J. E. Hopcroft and J. D. Ullman, "Introduction to Automata Theory, Language and Computations," Addison–Wesley, Reading, Mass. 1979.
6. R. M. Karp, Reducibility among combinatorial problems, *in* "Complexity of Computer Computations" (R. E. Miller and J. W. Thatcher, Eds.), Plenum, New York, 1972.
7. R. Ladner and N. Lynch, Relativization of problems about log space computability, *Math. Systems Theory* **10** (1976), 19–32.
8. N. A. Lynch, A. R. Meyer, and M. J. Fischer, Relativization of the theory of computational complexity, *Trans. Amer. Math. Soc.* **220** (1976), 243–287.
9. C. Rackoff, Relativized questions involving probabilistic algorithms, *in* "Procedings, 10th STOC, 1978," pp. 338–342.
10. J. I. Seiferas, M. J. Fischer, and A. R. Meyer, Separating nondeterministic time complexity classes, *J. Assoc. Comput. Mach.* **25** (1978), 146–167.