INFORMATION AND CONTROL **65**, 158–181 (1985)

Sparse Sets in *NP-P*: *EXPTIME* versus *NEXPTIME*[*]

J. HARTMANIS

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

N. IMMERMAN

Departments of Mathematics and Computer Science, Yale University, New Haven, Connecticut

AND

V. SEWELSON[†]

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

Received March 6, 1984; accepted May 9, 1985

This paper investigates the structural properties of sets in *NP-P* and shows that the computational difficulty of lower density sets in *NP* depends explicitly on the relations between higher deterministic and nondeterministic time-bounded complexity classes. The paper exploits the recently discovered *upward separation* method, which shows for example that there exist sparse sets in *NP-P* if and only if $EXPTIME \neq NEXPTIME$. In addition, the paper uses relativization techniques to determine logical possibilities, limitations of these proof techniques, and exhibits one of the first natural structural differences between relativized *NP* and *CoNP*.
© 1985 Academic Press, Inc.

INTRODUCTION AND OUTLINE OF RESULTS

It is well known that if higher deterministic and nondeterministic complexity classes do not collapse then the corresponding lower classes are also distinct. This *downward separation* is easily obtained by padding arguments and it yields, for example, that $EXPTIME \neq NEXPTIME$ implies $P \neq NP$. In this paper we prove that for some important complexity classes there also exists an *upward separation method* which reveals that certain structural properties of *P*, *NP*, and *PSPACE* would imply the separation of

* This research was supported, in part, by National Science Foundation Grants MCS83-01766 and MCS81-05754.

† Current address: Department of Mathematics and Computer Science, Dartmouth College, Hanover, NH 03755.

higher (exponential) deterministic and nondeterministic complexity classes.We investigate this strong and unexpected coupling of the structural properties of lower and higher complexity classes by the upward separation method and use relativization techniques to explore possible structural properties of NP sets as well as limitations of these proof techniques.

The original motivation for this work was the desire to understand better what makes the computational solution of problems in NP hard, provided $P \neq NP$. It is generally conjectured that $P \neq NP$ (Aho, Hopcroft, and Ullman, 1974; Garey and Johnson, 1979) and therefore, for example, it is thought to be very difficult to determine whether Boolean formulas in conjunctive normal form have satisfying assignments. Furthermore, there is a more explicit belief that the computational difficulty of finding satisfying assignments for Boolean formulas does not depend only on the existence of the aggregate of satisfiable Boolean formulas, SAT, but that there are "individual" instances of formulas for which it is hard to find satisfying assignments. In particular, we conjecture that there are syntactically simple, sparse subsets of Boolean formulas for which it cannot be decided in polynomial time whether they are satisfiable.

In this paper we show that lower density sets in NP and $PSPACE$ (but apparently not in $CoNP$) can be coded more compactly to yield denser sets in the correspondingly higher complexity classes. Therefore, lower density sets can exist in NP-P if and only if the higher complexity classes do not collapse.

More precisely, a set A is sais to be *sparse* if and only if it contains only a polynomial-in-n number elements up to size n (Berman and Hartmanis, 1977; Hartmanis, 1983)

Some of the following results were first outlined in (Hartmanis, 1983; Hartmanis, Immerman, and Sewelson, 1983).

THEOREM. *There exists a sparse set in NP-P, $PSPACE$-NP, or $PSPACE$-P if and only if, respectively, $NEXPTIME \neq EXPTIME$, $EXPSPACE \neq NEXPTIME$, and $EXPSPACE \neq EXPTIME$.*

This result has the interesting implication that if $P \neq NP$ but $EXPTIME = NEXPTIME$ then any sparse set in NP is already in P. Furthermore, since $A \in NEXPTIME - EXPTIME$ implies that $TALLY(A)$ is in NP-P (Book, 1974; Book, Wrathall, Selman, and Dobkin, 1978), we conclude that there exist sparse sets in NP-P if and only if there exist tally sets in NP-P. Later in this paper we will see that this property does not necessarily hold for $CoNP$-P.

It is interesting to note that it has been shown by Wilson (1980; Book, Wilson, and Xu, 1981) that there exist oracle sets A such that

$$P^A \neq NP^A \quad \text{but} \quad EXPTIME^A = NEXPTIME^A.$$

Furthermore, it has been shown more recently by Kurtz (1985) that there are oracle sets B such that $P^B \neq NP^B$ and NP^B-P^B contains no sparse sets. Since our results hold for relativized computations as well, we now know that $EXPTIME^A = NEXPTIME^A$ if and only if there are no sparse sets in NP^A-P^A. This shows that the two relativization results mentioned above are equivalent.

The above results can be generalized to show that the collapse of deterministic and nondeterministic classes below exponential time forces the corresponding sets of higher density from NP into P and that the collapse starts bounding the computation time of SAT.

THEOREM. *There are no $\delta(n) = n^{\log n}$ uniformly distributed dense sets in NP-P if and only if*

$$\bigcup_{c>0} NTIME[2^{c\sqrt{n}}] = \bigcup_{c>0} TIME[2^{c\sqrt{n}}]$$

and if this happens then $SAT \in TIME[2^{c\sqrt{n}}]$.

An interesting question is whether the assumption that SAT can be computed in less than exponential time has direct structural implications on the higher complexity classes. We show that at least for relativized computations the computation speed of NP problems can be decoupled from structural properties of the exponential complexity classes.

THEOREM. *There exists an oracle A such that*

$$NP^A \subseteq \bigcup_{c>0} TIME^A[n^{c \log n}]$$

and

$$EXPTIME^A \neq NEXPTIME^A.$$

These upward separation results have other implications about the relations between complexity classes based on the fact that standard diagonalization methods can be "slowed down" to yield sparse sets which separate complexity classes. We state one such result.

COROLLARY. *If $TIME[n^{\log n}] \subseteq NP$ then*

$$P \neq NP, \qquad EXPTIME \neq NEXPTIME,$$

$$EEXPTIME \neq NEEXPTIME, \text{ etc.,}$$

as well as

$$P \neq PSPACE \quad \text{and} \quad EXPTIME \neq EXPSPACE,$$

where $EEXPTIME = \bigcup_{c>0} TIME[2^{c2^n}]$.

Conversely, if we could separate P from NP by a sparse set then we would have shown that $P \neq NP$ as well as $EXPTIME \neq NEXPTIME$, which may be a much harder task than just showing that $P \neq NP$. Therefore, we conjecture that the separation of P and NP will be first acheived by an indirect proof or a constructive proof yielding a set in NP-P which is not sparse. For related ideas see (Kozen, 1978; Kozen and Machtey, 1980).

In this connection it is interesting to observe that until now the separations of nondeterministic complexity classes have been obtained with proofs by contradiction (translation lemmas), not by explicitly yielding the sets which separate the complexity classes (Cook, 1973; Ibarra, 1972; Seiferas, 1977). Furthermore, our results show that Ladner's (1975) delayed diagonalization construction of an incomplete set in NP-P, under the assumption that $P \neq NP$, cannot be modified, even by techniques that do not relativize, to yield a sparse set unless $EXPTIME \neq NEXPTIME$.

A very interesting open problem is the existence of sparse sets in the polynomial-time hierachy, PH (Garey and Johnson, 1979; Stockmeyer, 1976), under the assumption that $EXPTIME = NEXPTIME$. Or equivalently, if there are no sparse sets in NP-P can there exist sparse sets in PH-P? Quite surprisingly, we show that for some relativized computations the collapse $EXPTIME = NEXPTIME$ does not imply the collapse of the exponential-time hierarchy, $EXPH$, and therefore even though there are no sparse sets in NP-P there can be sparse sets in PH-P.

THEOREM. *There exists an oracle* A *such that* $EXPTIME^A = NEXPTIME^A$ *and* $\sum_2^{E,A} \neq NEXPTIME^A$. *Therefore there are no sparse sets in* $NP^A - P^A$ *but there are sparse sets in* $\sum_2^{P,A} - P^A$.

In spite of this oracle, we still conjecture that the collapse of $EXPTIME = NEXPTIME$ implies the collapse of the entire exponential-time hierarchy. However, the existence of the oracle gives evidence that the proof of such a result would be difficult.

Another relativization result reveals a limitation of the upward separation method and shows an early structural difference between relativized NP and $CoNP$ sets.

We first observe that the upward separation results for NP relativize and we get the following generalization.

COROLLARY. *For any A, there are sparse sets in NP^A-P^A if and only if there are tally sets in NP^A-P^A and this happens if and only if $EXPTIME^A \neq NEXPTIME^A$.*

Quite surprisingly this is not the case for relativized *CoNP* computations from which we can force out tally sets without forcing out all sparse sets

THEOREM. *There exists an oracle B such that $CoNP^B$-P^B contains sparse sets but $CoNEXPTIME^B = EXPTIME^B$ and therefore there are no tally sets in $CoNP^B$-P^B.*

SPARSENESS RESULTS

In this section we introduce the upward separation method and derive necessary and sufficient conditions for the existence of sparse sets in *NP-P*, *PSPACE-NP*, and *PSPACE-P*, respectively. We assume that the reader is familiar with the standard results and notation about deterministic and nondeterministic polynomial-time computations (Aho *et al.*, 1974; Garey and Johnson, 1979). We say that a set S, $S \subseteq \Sigma^*$, is *sparse* if and only if there exists a constant k such that

$$|S \cap (\varepsilon + \Sigma)^n| \leqslant n^k + k$$

i.e., the number of elements in S up to size n is polynomially bounded in n. Let P and NP denote the deterministic and nondeterministic polynomial time acceptable languages, respectively. Let

$$EXPTIME = \bigcup_{c>0} TIME[2^{cn}],$$

$$NEXPTIME = \bigcup_{c>0} NTIME[2^{cn}],$$

$$EEXPTIME = \bigcup_{c>0} TIME[2^{c2^n}],$$

and

$$NEEXPTIME = \bigcup_{c>0} NTIME[2^{c2^n}].$$

It is interesting to note that the location of the constant in the exponent is required by the upward separation method.

Intuitively, a sparse set has so few elements that it seems reasonable that there should be some compact way of naming its members such that we can recover the original string from this short name in nondeterministic exponential time. In fact, there is such a way and it is that property on which the following proof depends. We would like to be able to name a string by its position in the sparse set, e.g., the 15th string, but it would take double exponential time to recover the original string from this encoding. Hence, we shall use nondeterminism to be able to simulate knowing a string's position in S and hence be able to recover the string in nondeterministic exponential time.

THEOREM 1. *There exists a sparse set S in NP-P if and only if*

$$EXPTIME \neq NEXPTIME.$$

Proof. If $EXPTIME \neq NEXPTIME$ then we may assume that there exists a set A, $A \subseteq \{0, 1\}^*$,

$$A \in NEXPTIME - EXPTIME.$$

If we prefix each string in A by a 1 and interpret these strings as binary representations of integers, then we can convert A to tally notation:

$$TALLY(A) = \{1^n \mid n \in 1A\}.$$

Since A is in *NEXPTIME* and not in *EXPTIME*

$$TALLY(A) \in NP\text{-}P.$$

Thus we see that under this assumption there exists a sparse set in *NP-P* (Book, 1974; Book *et al.*, 1978).

Conversely, suppose that $EXPTIME = NEXPTIME$. We shall then show that every sparse *NP* set is in *P*. If we let S be a sparse set in *NP*, as stated above, we will encode S into a new set S' that we shall show to be in *NEXPTIME* and hence in *EXPTIME*. From this we shall be able to conclude that $S \in P$. Let $p(n)$ be a polynomial such that $|\Sigma^n \cap S| \leqslant p(n)$. Assume that the elements of S are lexicographically ordered. We define S' as follows.

$$S' = \{n \# i \# j \# k \# d \mid \exists x_1 < x_2 < \cdots < x_i \leqslant x < y_1 < y_2 < \cdots y_j \in S$$
$$\text{and} \quad |x_1| = |y_j| = n \text{ and the } k\text{th digit of } x \text{ is } d\}.$$

So, given x of length n in S, x corresponds to five-tuples $n \# i \# j \# k \# d$, where i and j encode lower bounds on the position in S of x and where k and d encode the characters that make up x. It is important to note three things at this point:

1. Given $x \in S$ of length n the $n \# i \# j \# k \# d$'s that correspond to x are of length $O(\log n)$ due to the sparseness of S.

2. Given a five-tuple $n \# i \# j \# k \# d$ we need to guess only a polynomial in n number of strings of length n to verify that $n \# i \# j \# k \# d \in S'$.

3. Verification of membership in S is a nondeterministic polynomial-in-n process—so requires an existential quantifier.

From these we see that $S' \in NEXPTIME$. Since we assumed that $EXPTIME = NEXPTIME$, we now know that $S' \in EXPTIME$. We shall use this fact to show that $S \in P$.

Given x of length n, we first find the exact census of S, i.e., we find $c_n = |\Sigma^n \cap S|$. We run our exponential time algorithm for membership in S' on the $p(n)$ pairs of strings

$$\begin{array}{ccc} n \# 1 \# 0 \# 1 \# 0 & \text{and} & n \# 1 \# 0 \# 1 \# 1, \\ n \# 2 \# 0 \# 1 \# 0 & \text{and} & n \# 2 \# 0 \# 1 \# 1, \\ n \# 3 \# 0 \# 1 \# 0 & \text{and} & n \# 3 \# 0 \# 1 \# 1, \\ & \dots & \\ n \# p(n) \# 0 \# 1 \# 0 & \text{and} & n \# p(n) \# 0 \# 1 \# 1. \end{array}$$

c_n will be the $i-1$, where i is the least integer such that both $n \# i \# 0 \# 1 \# 0 \notin S'$ and $n \# i \# 0 \# 1 \# 1 \notin S'$. This is because asking if a pair of strings $n \# i \# 0 \# 1 \# 0$ and $n \# i \# 0 \# 1 \# 1$ is in S' is simply asking if there are at least i strings of length n in S. Since we run an exponential in $\log n$ algorithm at most a polynomial in n number of times, we can find c_n in polynomial time.

Knowing the census c_n we observe that there is some pair of integers i_0 and j_0 whose sum is c_n such that there are exactly i_0 strings in S of lengh less than or equal to n that are smaller than x and exactly j_0 strings in S of length n that are greater than x. By noting that all of the $n \# i_0 \# j_0 \# k \# d$ in S' will describe x, we simply run our exponential time algorithm for S' on

$$\{n \# i \# j \# k \# d \mid i + j = c_n, k = 1, 2, \dots, n \text{ with } d \text{ the appropriate character of } x\}.$$

x will be in S if and only if for some pair i_0 and j_0 all of the appropriate $n \# i_0 \# j_0 \# k \# d$ are in S'. Again we run an exponential in $\log n$ algorithm at most a polynomial in n number of times and hence we can determine whether x is in S in polynomial time. ∎

By similar reasoning we can derive related results for $PSPACE$.

COROLLARY 2. *There are sparse sets in PSPACE-NP and PSPACE-P if and only if EXPSPACE$\neq$NEXPTIME and EXPSPACE$\neq$EXPTIME, respectively.*

Quite surprisingly, we will show later that related results do not hold for relativized *CoNP* computations, whereas all above results hold for relativized computations.

It is interesting to note that it is possible to relativize computations so that $NP^A \neq P^A$ and $EXPTIME^A = NEXPTIME^A$ (Wilson, 1980). Since Theorem 1 relativizes, this is equivalent to the case in which there are no sparse sets in NP^A-P^A but yet $EXPTIME^A = NEXPTIME^A$ (Kurtz, 1985).

Next we show that the previous result can be sharpened considerably. We say that a set S is *polynomial-time printable* if and only if there is a k_0 such that all the elements of S, up to size n, can be printed by a deterministic machine in time $n^{k_0} + k_0$. Clearly, every polynomial-time printable set is sparse and is in P.

COROLLARY 3. *There exists a polynomial-time printable set S such that $S \cap SAT \in NP\text{-}P$ if and only if EXPTIME$\neq$NEXPTIME.*

Proof. From the previous result we know that if there exists a sparse set in *NP-P* then *EXPTIME$\neq$NEXPTIME.* Therefore, we just have to show that

$$EXPTIME \neq NEXPTIME$$

implies that the desired set S exists and that

$$S \cap SAT \in NP\text{-}P.$$

Let $A \in NEXPTIME - EXPTIME$ then $TALLY(A) \in NP\text{-}P$. Since $TALLY(A)$ is in NP it can be reduced to SAT by a one-to-one, length increasing polynomial time reduction g (Berman and Harmanis, 1977; Cook, 1971; Aho *et al.*, 1974). This guarantees that

$$g[TALLY(A)] \quad \text{and} \quad g(1^*)$$

are sparse sets and furthermore that $g(1^*)$ is polynomial-time printable. Since g is a reduction of $TALLY(A)$ to SAT we know that

$$x \in TALLY(A) \Leftrightarrow g(x) \in SAT.$$

Therefore $g[TALLY(A)] \subseteq SAT \cap g(1^*)$ and if $g(1^i) \in SAT$ then $1^i \in TALLY(A)$, yielding $g[TALLY(A)] = SAT \cap g(1^*)$. Finally, $g[TALLY(A)] \in NP\text{-}P$ since $TALLY(A) \in NP\text{-}P$. This completes the proof. ∎

The existence of sparse polynomial-time recognizable sets S such that $S \cap SAT \in NP\text{-}P$ was conjectured by D. Joseph (1982). The above results show that sparse sets exist in $NP\text{-}P$ if and only if there exist polynomial-time printable sets S such that $S \cap SAT \in NP\text{-}P$.

COROLLARY 4. *There exist sparse sets in NP-P if and only if there exist tally sets in NP-P.*

Finally, it is easily seen that we have actually shown that $EXPTIME = NEXPTIME$ if and only if every sparse set in NP is polynomial-time printable. Therefore if one could show that there are sparse sets in P which are not polynomial-time printable it would follow not only that $P \neq NP$ but also that $EXPTIME \neq NEXPTIME$.

We have seen that there are polynomially sparse sets in $NP\text{-}P$ if and only there is a separation of complexity classes at the exponential level. It is interesting to ask whether something happens to other classes besides polynomial ones if $EXPTIME = NEXPTIME$, or whether there is some special relationship between exponential and polynomial classes. In the next result, we see that not only does the collapse of $EXPTIME$ and $NEXPTIME$ push all sparse from NP into P but it pushes sets of density $\delta(n)$ from $NTIME[\delta(n)]$ into $TIME[\delta(n)]$.

We say that the set A has density $\delta(n)$ if

$$|A \cap (\varepsilon + \Sigma)^n| \leqslant \delta(n).$$

COROLLARY 5. *For $n < \delta(n) \leqslant 2^n$ with $\log(\delta(n))$ computable in time $\delta(n)^c$ for some constant c, there is a set of density $\delta(n)$ in*

$$\bigcup_{c>0} NTIME[\delta(n)^c] - \bigcup_{c>0} TIME[\delta(n)^c]$$

if and only if EXPTIME $\neq$ NEXPTIME.

Proof. The proof is very similar to that of Theorem 1. If $EXPTIME \neq NEXPTIME$ and $A \in NEXPTIME - EXPTIME$ then

$$\{x \# 1^{\delta^{-1}(2^{|x|}) - |x|} \mid x \in A\} \in \bigcup_{c>0} NTIME[\delta(n)^c] - \bigcup_{c>0} TIME[\delta(n)^c],$$

and has density $\delta(n)$.

Conversely, if S has density $\delta(n)$ and

$$S \in \bigcup_{c>0} NTIME[\delta(n)^c] - \bigcup_{c>0} TIME[\delta(n)^c],$$

then the same S' as in Theorem 1 will be in $NEXPTIME - EXPTIME$. One has only to note that in this case the length of a five-tuple, $n \# i \# j \# k \# d$, in S' has length $O(\log(\delta(n)))$. ▌

OTHER DENSITIES

From the above, we see that in order to separate *EXPTIME* and *NEXPTIME* it is enough to have a sparse set in *NP-P*. We looked at sparse sets in order to formalize the notion of an "individual" instance of a satisfiable Boolean formula. Therefore, it seems that the choice of sets of polynomial density was rather arbitrary. We shall now investigate the consequences of the existence in *NP-P* of sets of various densities. We divide this investigation into two parts, one for sets whose density is greater than polynomial and one for sets whose density is smaller. The reason behind this division is that the technique of Theorem 1, in order for the original restricted density set S to be in P, requires that we guess and verify the membership of at most polynomially many strings. To do this, given a string x of length n, we need to find an interval around x that contains at most a polynomial in n number of strings of S. If S is of density greater than polynomial, it is not clear how to do this. And this seeming inability to trap polynomially many elements of S leads to the "uniformity conjecture" below. However, if S has density less than polynomial, then we can trap polynomially many elements of S around x by simply taking all strings in S up to length n.

For sets with density lower than polynomial in *NP-P* we are able to prove the separation of deterministic and nondeterministic complexity classes higher than exponential.

THEOREM 6. *For any monotonically increasing, time-constructible $T(n)$, where $T(n) > 2^n$ and where $2^{T^{-1}(n)}$ is computable in time bounded by a polynomial in n,*

$$\bigcup_{c>0} NTIME[T(n)^c] \neq \bigcup_{c>0} TIME[T(n)^c]$$

if and only there is a set of density $2^{T^{-1}(n)}$ in NP-P.

Proof. Let $T(n) > 2^n$ be monotonically increasing and time-constructible and suppose that

$$\bigcup_{c>0} NTIME[T(n)^c] \neq \bigcup_{c>0} TIME[T(n)^c].$$

Let $A \in \bigcup_{c>0} NTIME[T(n)^c] - \bigcup_{c>0} TIME[T(n)^c]$. It is easy to see that $A' \in NP - P$, where $A' = \{x \# 1^{T(|x|)-|x|} \mid x \in A\}$. To see that A' has the correct density, note that up to length n there are at most $2^{T^{-1}(n)}$ strings of the form $x \# 1^{T(|x|)-|x|}$, since for these strings $|x| = T^{-1}(n)$.

Conversely, if we assume that

$$\bigcup_{c>0} NTIME[T(n)^c] = \bigcup_{c>0} TIME[T(n)^c]$$

and let $S \in NP$ and suppose S has density $2^{T^{-1}(n)}$, we shall show that $S \in P$. Let S be in $NTIME[n^k + k]$ for some k. Define S' to be

$$\{t \# i \mid \text{up to length } T(\log t) \text{ there are at least } i \text{ strings in } S\}.$$

Note that for $t \# i$ to be in S', the density of S ensures that $i < t$. $S' \in NTIME[T(n)^c]$ for some c, since on input $t \# i$ of length at most $2 \log t$ we guess $i < t$ strings of length $T(\log t)$ and verify that they are in S. This takes time

$$iT(\log t)[T(\log t)^k + k] \leqslant tT(\log t)[T(\log t)^k + k].$$

Since $T(n) > 2^n$, this takes time at most

$$T(\log t)T(\log t)[T(\log t)^k + k] \leqslant T(2 \log t)T(2 \log t)[T(2 \log t)^k + k],$$

which is polynomial in T of the input length, as desired.

Since we assumed that

$$\bigcup_{c>0} NTIME[T(n)^c] = \bigcup_{c>0} TIME[T(n)^c],$$

then $S' \in TIME[T(n)^d]$ for some d. Hence, on input $t \# i$ we can deterministically compute the maximal i_t such that $t \# i_t \in S'$. But then, in $TIME[T(n)^f]$ we can compute for input $t \# 1$ the sequence of i_t stings in S up to size $T(\log t)$, $x_1 \# x_2 \# \cdots \# x_{i_t}$. Thus, for any x of length n, a deterministic polynomial time machine can compute the same string from input $2^{T^{-1}(n)} \# 1$ and check if $x \in S$. Hence, if

$$\bigcup_{c>0} NTIME[T(n)^c] = \bigcup_{c>0} TIME[T(n)^c]$$

then $S \in P$. ▌

In particular, if we let $T(n) = 2^{2^n}$ in Theorem 6 we get the following corollary. Recall that a set A, $A \subseteq \Sigma^*$, is *super sparse* if there exists a constant k such that

$$|A \cap (\varepsilon + \Sigma)^n| \leqslant k \log n.$$

COROLLARY 7. *There exist super-spare sets in NP-P if and only if*

$$EETIME = \bigcup_{r>0} TIME[2^{r2^n}] \neq \bigcup_{r>0} NTIME[2^{r2^n}] = NEETIME.$$

The sparseness results can also be generalized to sets of greater than polynomial density and the corresponding collapse of deterministic and nondeterministic classes below exponential time. Furthermore, the collapse of these classes starts bounding the computation time of *SAT*. We illustrate these possible generalizations with the next result.

Recall that the set A has density $\delta(n)$ if

$$|A \cap (\varepsilon + \Sigma)^n| \leqslant \delta(n).$$

To prove the following result we need the assumption that the $\delta(n)$-dense sets are uniformly distributed. We conjecture that this assumption is not needed, but so far have not been able to eliminate it from the proof:

> A set A of density $\delta(n)$ is *uniformly distributed* if and only if any interval of length $2^n/\delta(n)$ contains at most polynomially many elements of A up to size n, where an *interval* is any set of strings consecutive in the lexicographic ordering of Σ^*.

Note that for A to be uniformly distributed, it is enough for each of the $\delta(n)$ canonical intervals that divide Σ^n equally to have only polynomially many elements of A. A canonical interval consists of all strings of length n that have the same $\log(\delta(n))$ leading bits.

THEOREM 8. *There are no $\delta(n) = n^{\log n}$ uniformly distributed dense sets in NP-P if and only if*

$$\bigcup_{c>0} NTIME[2^{c\sqrt{n}}] = \bigcup_{c>0} TIME[2^{c\sqrt{n}}]$$

and if this happens then $SAT \in TIME[2^{c\sqrt{n}}]$.

Proof. To prove one direction, suppose that

$$\bigcup_{c>0} NTIME[2^{c\sqrt{n}}] \neq \bigcup_{c>0} TIME[2^{c\sqrt{n}}],$$

and let $A \subseteq \{0, 1\}^*$ and

$$A \in \bigcup_{c>0} NTIME[2^{c\sqrt{n}}] - \bigcup_{c>0} TIME[2^{c\sqrt{n}}].$$

Define

$$A' = \{x \# 1^{2^{\sqrt{|x|}} - |x|} \mid x \in A\}.$$

Clearly $A' \in NP\text{-}P$. Given

$$w = x \# 1^{2^{\sqrt{|x|}} - |x|}$$

of length n, $|x| = \log^2 n$. Therefore, there are at most $2^{\log^2 n}$ such w, and so A' must have density $\leqslant n^{\log n}$. Each of the canonical intervals has at most one element of A', since A' has at most one string of length n beginning with any sequence of $\log^2 n$ bits. Thus A' is uniformly distributed.

Conversely, suppose $S \in NP$ has density $\delta(n) = n^{\log n}$ and that each canonical interval has fewer than $n^{k'} + k'$ elements of S. Then the set

$C = \{t \# l \# i \mid$ in the lth interval of strings of length t there are at least i elements in $S\}$

is in $NTIME\ [2^{k\sqrt{n}}]$, for some k. Since for x of length t, the representation of t, i, and l is bounded by $\log^2 t$, and in time $t^k = 2^{k \log^2 t}$, we can guess the $i \leqslant t^{k'} + k'$ strings and verify that they are in the lth interval and in S. A string x is in the lth interval if and only if its first $\log(\delta(n))$ bits are l, hence it is easy to verify.

If $NTIME[2^{k\sqrt{n}}] \subseteq TIME[2^{c\sqrt{n}}]$ then in $TIME[2^{d'\sqrt{n}}]$ we can compute the maximal $i_{t,l}$ such that $t \# l \# i_{t,l} \in C$. This gives the number of strings of S of length t in the lth interval. But then, a $TIME[2^{d''\sqrt{n}}]$ machine can compute, for input $t \# l$, the $i_{t,l}$ strings in S in the lth interval. Hence, for any x of length t, a deterministic polynomial time machine can compute the strings of S of length n in the same interval as x and check if $x \in S$. Thus $S \in P$. To see how the computation time of SAT is affected, observe that $NP \subseteq \bigcup_{c>0} NTIME[2^{c\sqrt{n}}]$. ∎

The crucial points that make the above technique work are that when we encode the $\delta(n)$ dense set into shorter strings, we encode strings of length n into strings of length $\log(\delta(n))$ and that to verify the encoded set, we guess no more than $n^k + k$ strings for some k. This is where the uniformity was required.

In the above proof we used the number of the canonical interval for the $\log(\delta(n))$ bits. However this can be generalized to any $\log(\delta(n))$ bits which somehow encode the endpoints of an interval with only polynomially many elements of the set in question. From this we propose the following weaker notion of uniform distribution. A $\delta(n)$ dense NP set S is *uniformly distributed* if given a string x of length n, we can guess $\log(\delta(n))$ bits of information from which in time $n^k + k$, we can nondeterministically compute an interval that contains only polynomially many elements of S, including x, if

$x \in S$. It is our belief and we conjecture that any $\delta(n)$ dense *NP* set S is uniformly distributed. The reason for this conjecture is that an *NP* set is computationally fairly simple. It would seem incongruous for this simplicity to be coupled with a complex distribution. If this uniformity conjecture is true, not only may we drop the uniformity condition from the above theorem, but we have an interesting connection between *NP* sets and Kolmogorov complexity with bounded computation time.

In the last theorem, we saw how structural properties of *NP* sets affect the deterministic computation of *SAT*. This leads us to wonder if the computation time of *SAT* has any structural effect on *NP*. A negative answer to this question is suggested by the following theorem. As with all relativization arguments, this does not imply that this holds for regular (not relativized) computations, but it shows that there are "worlds" for which this result holds and indicates that the result is most likely very hard to prove or disprove for regular computations.

THEOREM 9. *There exists an oracle A such that*

$$NP^A \subseteq TIME^A[n^{\log n}] \text{ and } EXPTIME^A \neq NEXPTIME^A.$$

Proof Sketch (For details see Sewelson, 1983). This oracle is constructed as the disjoint union of two sets. The strings of A that start with a 0 encode a $NEXPTIME^A$ set into A that can not be decoded in exponential time using the oracle A thus ensuring that $NEXPTIME^A \neq EXPTIME^A$. The strings of A that start with 1 encode an NP^A-complete set into A such that this complete set can be recovered in $n^{c \log n}$-time using the oracle A thus ensuring that $NP^A \subseteq \bigcup_{c>0} TIME^A[n^{c \log n}]$. More precisely,

1. $L = \{1^n \mid \exists y(|y| = 2^n \text{ and } 0y \in A)\}$ will be constructed by diagonalization so that it is not in $EXPTIME^A$.

2. $L(N^A) = \{x \mid 1 \# x \# 0^{n^{\log n}} \in A\}$, where $L(N^B)$ is complete for NP^B for any oracle B and where N^B runs in time $n^k + k$.

A is constructed in stages so that the above two requirements are met. ▌

SEPARATION RESULTS

Next we show that the upward separation results have very strong implications under the assumption that *NP* contains a deterministic, time bounded complexity class above polynomial. If this happens, then standard diagonalization arguments, when they are slowed down, can produce sparse and super-sparse sets in *NP-P*. As we have seen, this forces the higher complexity classes not to collapse.

THEOREM 10. *Let $R(n)$ be real-time computable and for all $k \geqslant 1$ let*

$$\lim_{n \to \infty} \frac{n^k}{R(n)} = 0.$$

Then $TIME[R(n)] \subseteq NP$ implies that for any monotonically increasing, time-constructible $T(n)$, where $T(n) > 2^n$ and where $2^{T^{-1}(n)}$ is computable in time polynomial in n,

$$\bigcup_{c>0} NTIME[T(n)^c] \neq \bigcup_{c>0} TIME[T(n)^c].$$

Proof. Since $R(n)$ is real-time computable the limit condition permits us to diagonalize over all deterministic polynomial time machines. Since this diagonalization process can be stretched out by rejecting large numbers of elements before diagonalizing over the next machine, we see that deterministic diagonalization can yield sets of any computable density in $TIME[R(n)] - P$. Therefore, if

$$TIME[R(n)] \subseteq NP,$$

we know that $NP \neq P$ and because of the arbitrarily sparse sets in NP-P an application of Theorem 6 separates deterministic and nondeterministic classes at the $T(n)^c$ level. ▮

Taking particular values of $R(n)$ and $T(n)$ yields

COROLLARY 11. *If $TIME[n^{\log n}] \subseteq NP$ then*

$$P \neq NP,\ EXPTIME \neq NEXPTIME,\ EEXPTIME \neq NEEXPTIME,$$

etc., as well as

$$P \neq PSPACE,\ EXPTIME \neq EXPSPACE, \qquad \text{etc.}$$

Theorem 10 can easily be generalized to other complexity classes.

COROLLARY 12. *If $TIME[n^{\log n}] \subseteq PSPACE$ then*

$$P \neq PSPACE,\ EXPTIME \neq EXPSPACE, \qquad \text{etc.}$$

It is interesting to note that if we could show that $P \neq NP$ by any process which can be "stretched" to yield sparse and super sparse sets, then we would not only have shown that $P \neq NP$, but also that

$$EXPTIME \neq NEXPTIME$$

and

$$EEXPTIME \neq NEEXPTIME.$$

Clearly, one possibility of showing that not only $P \neq NP$ but that the higher deterministic and nondeterministic time computations are different, is suggested by Theorem 10. This would require showing that NP contains some (properly defined) deterministic time classes above polynomial. Unfortunately, at this time we do not believe that this is true. We conjecture that NP contains P properly and that furthermore NP contains no deterministic time classes above P. We conjecture that the corresponding relations hold between P and $PSPACE$ and NP and $PSPACE$.

In spite of this belief, we are able to find a relativized world where, indeed, NP contains a deterministic-time class higher than polynomial and hence one where all classes are distinct. It is interesting to contrast this oracle with the long standing unsuccessful search for a single oracle A such that A simultaneously separates every level of the polynomial time hierarchy, ie.

$$\forall k \Sigma_k^{p,A} \neq \Pi_k^{p,A}$$

(Baker and Selman, 1979).

THEOREM 13. *There is an oracle B such that*

$$\bigcup_{c>0} TIME^B[n^{c \log n}] \subseteq NP^B.$$

Proof Sketch (For details see Sewelson, 1983). We construct B to contain an encoded $\bigcup_{c>0} TIME^B[n^{c \log n}]$-complete set. Let M^A be an $n^{k \log n}$ time bounded oracle Turing machine such that $L(M^A)$ is complete for $\bigcup_{c>0} TIME^A[n^{c \log n}]$ under polynomial-time many–one reductions for all oracles A. If we create B such that

$$M^B(x) \text{ accepts} \Leftrightarrow \exists y(|y| = |x|^2,\ x \# y \in B)$$

then $L(M^B) \in NP^B$. Hence we shall have

$$\bigcup_{c>0} TIME^B[n^{c \log n}] \subseteq NP^B,$$

as desired. ∎

Sparse Sets in the Polynomial-Time Hierarchy

A fascinating and important open problem in this research area is the relation between the nonexistence of sparse sets in *NP-P* and in other parts of the polynomial-time hierarchy. The main problem is whether the collapse $EXPTIME = NEXPTIME$ which forces all sparse sets from *NP* into *P* also forces all sparse sets from the polynomial-time hierarchy (Garey and Johnson, 1979; Stockmeyer, 1976) into *P*.

The importance of this problem is emphasized by the fact that many interesting sparse sets are in the polynomial hierarchy if and only if the hierarchy is finite. For example, it has been observed by A. R. Meyer that there exist polynomial size circuits for *SAT* if and only if there exists a sparse oracle set *S* such that $SAT \subseteq P^S$ (Berman and Hartmanis, 1977). Furthermore, from (Karp and Lipton, 1980) we know that if such an oracle *S* exists then *S* is in the polynomial-time hierarchy and the hierarchy is therefore finite. If $EXPTIME = NEXPTIME$ would also force all sparse sets from the polynomial-time hierarchy into *P*, then the existence of polynomial-size circuits for *SAT* or, equivalently, the existence of a sparse complete-set for *NP* under polynomial-time Turing-reducibility, would imply that $P = NP$. The equivalence comes from the assumption that all sparse sets in the polynomial-time hierarchy are in *P*. For under this assumption when the polynomial-size circuits for *SAT* give us a sparse set *S* in the polynomial-time hierarchy such that $NP \subseteq P^S$, *S* is in *P* hence in *NP*.

We recall that the existence of sparse complete-sets under many-one polynomial-time reductions implies that $P = NP$ (Mahaney, 1980).

Another interesting open problem is whether $EXPTIME = NEXPTIME$ implies that for every sparse subset $S \subseteq SAT$, which we know is in *P*, we can find in polynomial time a satisfying assignment for *F* in *S*. If $EXPTIME = NEXPTIME$ forces all sparse sets from the polynomial time hierarchy into *P*, then we can easily determine the minimal satisfying assignment of *F* in *S* in polynomial time from a sparse set in Δ_2^P, which by assumption is in *P*.

To define the *exponential hierarchy*, *EXPH*, let

$$\textstyle\sum_1^E = NEXPTIME \qquad \text{and} \qquad \prod_1^E = CoNEXPTIME.$$

$\sum_2^E$ consists of all languages *C* such that there exist a constant *c* and a polynomial time predicate R_c for which

$$C = \{x \mid (\exists y, |y| \leqslant 2^{c|x|})(\forall z, |z| \leqslant 2^{c|z|})[R_c[x, y, z]]\};$$

the other classes are defined analogously.

Equivalently, we can define the *exponential hierarchy* as follows:

$$\textstyle\sum_0^E = EXPTIME \qquad \text{and} \qquad \forall k > 0 \textstyle\sum_k^E = NEXPTIME^{\Sigma_{k-1}^P}.$$

Thus, for example, $\sum_2^P = NP^{SAT}$ and $\sum_2^E = NEXPTIME^{SAT}$. Note that $\sum_{k+1}^E \neq \sum_k^E$ if and only if $\sum_{k+1}^P - \sum_k^P$ contains a sparse set. From this definition it is clear that a collapse of the polynomial hierarchy will imply a collapse of the exponential hierarchy. Since there is an oracle A such that $P^A \neq NP^A$ but $EXPTIME^A = NEXPTIME^A$ (Wilson, 1980), it becomes equally clear that it is possible for $EXPTIME = NEXPTIME$ but yet to have the exponential hierarchy exist at higher levels. The reason a collapse at the base of the polynomial hierarchy topples the entire structure is the close coupling of the oracle and the underlying machine, e.g., $\sum_2^P = NP^{NP}$. In light of this, it is the behavior of the polynomial hierarchy that is peculiar, and not that of the exponential hierarchy. We expect the exponential hierarchy to collapse if $EXPTIME = NEXPTIME$ only because the hierarchy with which we have the most familiarity has the accident of being a special case.

As the next result shows, there are oracles for which $EXPTIME^A = NEXPTIME^A$ but the exponential hierarchy does not collapse, thus not all the sparse sets in the polynomial hierarchy are forced into P^A. For related results see (Heller, to appear).

THEOREM 14. *There is an oracle A such that*

$$EXPTIME^A = NEXPTIME^A \qquad \text{but} \qquad \textstyle\sum_2^{E,A} \neq NEXPTIME^A.$$

Proof. This oracle is constructed as the disjoint union of two sets. The strings of A that start with a 0 encode a $\sum_2^{E,A}$ set into A that can not be decoded in exponential time using the oracle A thus ensuring that $\sum_2^{E,A} \neq EXPTIME^A$. The strings of A that start with a 1 encode an $NEXPTIME^A$-complete set into A such that this complete set can be recovered in exponential time using the oracle A thus ensuring that $EXPTIME^A = NEXPTIME^A$. More precisely,

1. $L = \{x \mid \exists y \forall z(|y| = 2^{|x|} \text{ and } |z| = |y| \rightarrow 0xyz \in A)\}$ will be constructed by diagonalization so that it is not in $EXPTIME^A$.

2. $L(N^A) = \{x \mid 1 \# x \# 0^{2^{k|x|}} \in A\}$, where $L(N^B)$ is complete for $NEXPTIME^B$ for any oracle B and where N^B runs in time 2^{kn}.

We construct A in stages so that the above two requirements are met. Before stage n, the membership in A of no strings of length greater than kn has been determined. We let $\{M_i^-\}$ be an enumeration of all exponential time bounded oracle machines, where M_i^- runs in time $2^{\log(i)n}$. We can get such an enumeration by simply taking a standard enumeration of all deterministic Turing machines and putting a 2^{kn} clock on the kth machine. Since each machine appears infinitely often, it will appear as some M_k where $2^{\log(k)n}$ is greater than its actual running time. Such an enumeration is needed since we wish to diagonalize over these machines and need the nth machine to query fewer than 2^{2^n} strings on an input of length n.

Initially, $A \neq \phi$. Stage n, Part 1. To guarantee condition 1, we find some string x_0 of length n such that for each y of length 2^n there is a z of length 2^n such that the membership in A or $\bar{A}$ of $0x_0yz$ has not been determined. For such an x_0 to exist, we must have determined the membership of fewer that 2^{2^n} strings so far.

To diagonalize over $EXPTIME^A$, we run $M_n^A(x_0)$ adding to $\bar{A}$ the strings that M_n^A queries whose membership had not yet been determined. This will guarantee that A remains consistent throughout the construction. Note that this determines at most $2^{n \log n}$ strings. If M_n^A accepted x_0 we want to put x_0 into $\bar{A}$. Hence, for each y of length 2^n we add $0x_oyz$ to $\bar{A}$ for some z of length 2^n. This can be done because there are fewer than 2^{2^n} strings in $\bar{A}$ so far. If M_n^A rejected x_0 we want to add x_0 to A. Hence, we must find some y_0 of length 2^n such that for no z of length 2^n has $0x_0y_0z$ been put into $\bar{A}$. We then add all $0x_0y_0z$ to A where $|z| = 2^n$. Such ay_0 exists since we have added fewer than 2^{2^n} strings to $\bar{A}$ so far.

It is important to note that:

1. We queried strings of length up to $2^{n \log n}$.
2. We put into $\bar{A}$ at most $2^{n \log n}$ queried strings.
3. We add to A strings of length $2^{n+1} + n + 1$.

Part 2. To guarantee condition 2, we run N^A on strings x of length between $(n-1)\log(n-1)$ and $n \log n$. We try to make N^A accept to prevent us from having to determine the membership of too many strings. To do this, we see what strings we can consistently add to A to make N^A accept. If this is possible, we add to A or $\bar{A}$ the queried strings along an accepting computation path. If we can not make N^A accept, then it does not matter what happens to A in the future, so we do nothing. We add $1 \# x \# 0^{2^{k|x|}}$ to A if $N^A(x)$ accepted and to $\bar{A}$ if it rejected.

It is important to note that:

1. We queried strings of length up to $2^{kn \log n}$.
2. We add to A or $\bar{A}$ at most $2^{kn \log n} 2^{n \log n}$ strings.
3. We add strings of length $2^{kn \log n} + n \log n + 1$, which is longer than any strings put into A or $\bar{A}$ by any previous process so that there are no conflicts. ∎

The above results, as well as other earlier work, seem to indicate that there may be a fundamental difference between many–one and Turing polynomial time reducibilities on sets in NP. For example, the existence of polynomial-size circuits for NP may not necessarily imply that $P = NP$ as does the existence of sparse many-one complete NP set (Mahaney, 1980).

In response to this question, very recently, Kurtz (1985) and, simultaneously, Immerman and Mahaney (1982), have shown that there exists an oracle A and a sparse oracle S such that

$$NP^A \neq P^A \qquad \text{but} \qquad NP^A \subseteq (P^A)^S.$$

This shows that for these oracles the Karp–Lipton result is indeed different from Mahaney's result (Mahaney, 1980; Karp and Lipton, 1980).

Clearly the strong assumption that the exponential hierarchy collapses to $EXPTIME$ or the even stronger assumption that $EXPTIME = EXPSPACE$ has interesting implications since if forces all sparse sets from PH into P.

COROLLARY 15. *If $EXPH = EXPTIME$ then the existence of polynomial-size circuits for NP and PSPACE implies, respectively, that $NP = P$ and $PSPACE = P$.*

Proof. From (Karp and Lipton, 1980) we know that the polynomial-size circuits for NP and $PSPACE$ are in PH and since their descriptions form sparse sets, the hypotheses guarantees that they are in P and therefore $NP = P$ and $PSPACE = P$, respectively. ∎

The upward separation method works well for NP and $PSPACE$ computations for which we can code down sparse sets based on *guessing*, *verifying*, and *counting*. At the same time, $CoNP$ computations do not have explicitly the ability to guess as do NP computations and therefore the upward separation method does not seem to apply to sparse sets in $CoNP$. Our next result shows that at least for some relativized computations this is indeed the case.

THEOREM 16. *There is an oracle A such that there are sparse sets in $CoNP^A$-P^A but there are no sparse sets in NP^A-P^A.*

Proof. We first set up some notation. Let $\{M_i\}$ be an enumeration of all polynonial time oracle machines where M_i runs in time $n^{\log i} + \log i$. We need this enumeration for a diagonalization which will require the running time of the nth machine to be less than 2^n. N is a nondeterministic exponential time oracle machine running in time 2^{kn}, where k is a constant, such that $L(N^A)$ is complete for $NEXPTIME^A$ for all oracles A.

This oracle is constructed as the disjoint union of two sets. The strings of A that begin with a 1 encode a sparse $CoNP^A$ set that is not in P^A. The strings of A that begin with a 0 encode an $NEXPTIME^A$ complete set that can be recovered in exponential time, thus ensuring that $EXPTIME^A = NEXPTIME^A$, i.e., that there are no sparse sets in NP^A-P^A. More precisely, we will construct A in stages so that

1. $S = \{x \mid \forall y \mid y \mid = \mid x \mid \rightarrow 1xy \notin A\} \in CoNP^A$-$P^A$ and is sparse.
2. $0 \# x \# 1^{2^{k|x|}} \in A$ iff $N^A(x)$ accepts.

Condition 1 will be achieved by diagonalization over polynomial time oracle machines.

We add elements to A in stages. Each stage has two parts, one for each condition. No elements are to be in A except those explicitly mentioned. Initially, A is empty.

Stage n, *Part* 1. To achieve condition 1, we diagonalize over P^A machines and put at most one string of each length into S, ensuring sparseness.

If $2^{k \log^2(n)} 2^{\log^2(n)} \log^2(n)n + n^{\log n} + \log n < 2^n$ (which is true almost everywhere) then find some x_0 of length n such that for no y of length n has the membership of $1x_0y$ been determined. Since the left-hand side of the above inequality is the number of strings of length $2n+1$ whose membership in A or $\bar{A}$ is determined (this will become clear after reading part 2 of this construction) and since there are $2^n x$'s of length n, we can find such an x_0.

Run M_n^A on input x_0 reserving for $\bar{A}$ all strings queried whose membership has not yet been determined—this will ensure that as we change A in future stages it will not affect this particular computation. The number of strings queried is at most the running time of M_n^A on x_0, $n^{\log n} + \log n$. If it accepts we want $x_0 \notin S$. So we add some $1x_0y$ to A. Since we put into $\bar{A}$ at most $n^{\log n} + \log n < 2^n$ strings, we can find a free $1x_0y$. If it rejects we want $x_0 \in S$. So we put into $\bar{A}$ all $1x_0y$.

Whether or not M_n accepted x_0, if $2^{k \log^2(n)} 2^{\log^2(n)} \log^2(n)n < 2^n$ we want S to remain sparse in spite of strings put into $\bar{A}$ in future stages. So, we shall make sure that the only strings in S are those put in by the above diagonalization. To do this, for each $x \neq x_0$ of length n add one $1xy$ to A_n guaranteeing that $x \notin S$. Please note that for every $x \neq x_0$ of length n there is some y of length n such that the membership of $1xy$ has not been determined, since the left-hand side of the above inequality is the number of strings of length $2n+1$ whose membership in A or $\bar{A}$ is already determined and we have 2^n choices for $1xy$. S will be sparse since the above inequality is true almost everywhere.

It is important to note that in the part 1's of the first n stages:

1. We queried strings of length up to $n^{\log n} + \log n$, the running time of M_n^A on an input of length n.

2. We put into $\bar{A}$ at most $n(n^{\log n} + \log n)$ queried strings, the number of stages times the running time of M_n^A.

3. We add to A strings of length up to $2n+1$.

Part 2. Run N^A on inputs x such that $\log^2(n) \leqslant |x| < \log^2(n+1)$. If $N^A(x)$ accepts, add $0 \# x \# 1^{2^{k|x|}}$ to A and put into $\bar{A}$ all strings queried on some accepting computation path whose membership in A had not yet been determined (again to preserve this computation even as we change A in future stages). If $N^A(x)$ rejects then to preserve this rejection we would have to add to $\bar{A}$ strings on all computation paths. This would be too many for our counting arguments so instead we see if we can, by adding to A, make it accept. If so, add those strings on one accepting computation path and $0 \# x \# 1^{2^{k|x|}}$ to A and put into $\bar{A}$ those strings queried along one accepting path whose membership in A had not yet been determined. If we can not make N^A accept, we need do nothing to A since nothing will change this acceptance.

It is important to note that in all part 2's so far:

1. We queried strings of lengths between $2^{k \log^2(n)}$ and $2^{k \log^2(n+1)}$, the running time of N^A.

2. We determine the membership of at most $2^{k \log(n+1)} 2^{\log^2(n+1)} \log^2(n+1)n$ strings, the running time of N^A times the number of strings of length between $\log^2(n)$ and $\log^2(n+1)$ (the number of strings of length $\log^2(n+1)$ times an upper bound on $\log^2(n+1) - \log^2(n)$) times the number of stages.

3. We add to A strings of length $2^{k \log^2(n+1)} + \log^2(n+1) + 3$.

Observe that in Part 1 we query strings shorter than those to be added in Part2 so Part 2 does not interfere. In addition, part 2 adds strings as it goes along that are larger that N^A could query so Part 2 does not interfere with itself. Since we put into $\bar{A}$ the strings we queried, the strings added in future Part 1's can not affect what is done here. ∎

COROLLARY 17. *There is an oracle A such that $CoNP^A$-P^A has sparse sets, but no tally sets.*

Proof. The oracle A of Theorem 16 suffices because $CoNP^A$-P^A contains sparse sets, but because $NEXPTIME^A = EXPTIME^A$ all tally sets in NP^A are in P^A. This also forces all tally sets from $CoNP^A$ into P^A, since if T is a tally set in $CoNP^A$-P^A then the tally set $1^* \cap \bar{T}$ is in NP^A and hence in P^A. But now T is in P^A, since $x \in T$ if and only if x of the form 1^k and x is not in $1^* \cap \bar{T}$. ∎

This theorem has many interesting implications. It is one of the first oracles, A, to display a structural difference between NP^A and $CoNP^A$. Not only does it show that NP^A and $CoNP^A$ can be distinguished by the existence of sparse sets, but that only for $CoNP^A$ can we decouple sparse sets from tally sets. Note that by the same methods we can show structural differences for relativized *CoNP* and *PSPACE*, since the upward separation method works for *PSPACE* (see Corollary 2). It also demonstrates that the proof technique of the first theorem is in some sense tight, since when the technique is applied to *CoNP*, we must go to $\sum_2^E$, not *CoNEXPTIME* to decode the encoded sparse set. And since $EXPTIME = NEXPTIME$ does not necessarily imply that $\sum_2^E = EXPTIME$ the *CoNP* analogue of the first theorem fails.

Acknowledgments

The authors would like to thank Ron Book, Deborah Joseph, Stuart Kurtz, Steve Mahaney, and Yaacov Yesha for helpful discussions and suggestions. The original search for the upward separation method was stimulated by Kurtz's relativization result (Kurtz, 1985) obtaining during the AMS Summer Institute on Recursive Function Theory at Cornell University in 1982.

Received March 6, 1984; accepted May 9, 1985

References

Aho, A. V., Hopcroft, J. E., and Ullman, J. D. (1974), "The design and Analysis of Computer Algorithms," Addison-Wesley, Reading, Mass.

Baker, T., Gill, J., and Solovay, R. (1975), Relativizations of the $PN = ?NP$ question, *SIAM J. Comput.* **4**, 431–442.

Book, R. V., Wrathall, C., Selman, A., and Dobkin, D. (1978), Inclusion Complete Tally Languages and the Hartmanis-Berman Conjecture, *Math. Systems Theory* **11**, 1–8.

Berman, L., and Hartmanis, J. (1977), On isomorphisms and density of *NP* and other complete sets, *SIAM J. Comput.* **6**, 305–327.

Book, R.V. (1974), Tally languages and complexity classes, *Infor. and Control* **26**, 186–193.

Baker, T., and Selman, A. (1979), A second step towards the polynomial hierarchy, *Theoret. Comput. Sci.* **8**, 177–187.

Book, R. V., Wilson, C., and Xu, M. (1981), Relativizing time and space *in* "IEEE Found. Comput. Sci. Symposium," 254–259.

Cook, S. A. (May 1971), The complexity of theorem-proving procedures *in* "Proceedings of the 3rd Annual ACM Symposium on the Theory of Computation," 151–158.

Cook, S. A. (1973), A hierarchy of nondeterministic time complexity, *J. Comput. System Sci.* **7**, 343–353.

Garey, M. R., and Johnson, D. S. (1979), "Computers and Intractability, A Guide to the Theory of *NP*-Completeness," Freeman, San Francisco.

Hartmanis, J. (1983), On sparse sets in *NP-P*, *Inform. Process. Lett.* **16** (1983), 55–60.

HARTMANIS, J., IMMERMAN, N. AND SEWELSON, V. (1983), Sparse sets in *NP-P*: *EXPTIME* versus *NEXPTIME in* "Proceedings 15th ACM Symposium on the Theory of Computation," 382–391.

HELLER, H., On relativized exponential and probabilistic complexity classes, *Inform. and Control*, to appear.

IBARRA, O. (1972), A note concerning nondeterministic tape complexities, *J. Assoc. Comput. Mach.* **19**, 608–612.

IMMERMAN, N., AND MAHANEY, S., Oracles for which *NP* has polynomial size circuits, draft, September 1982.

JOSEPH, D. (1982), Private communication.

KARP, R. M., AND LIPTON, R. J., Some connections between nonuniform and uniform complexity classes, *in* "Proceedings 12th Annual ACM Symposium on Theory of Computation," April 1980, 302–309.

KOZEN, D., AND MACHTEY, M., "On Relative Diagonals," IBM Research Report RC 8184, April 1980.

KOZEN, D. C. (1978), Indexings of subrecursive classes, *in* "Proceedings 10th Annual ACM Symposium on Theory of Computing," 287–295.

KURTZ, S. A. (1985), Sparse sets in *NP-P*: Relativizations, *SIAM J. Comput.* **14** (1985), 113–119.

LADNER, R. E. (1975), On the structure of polynomial time reducibility, *J. Assoc. Comput. Mach.* **22**, 155–171.

MAHANEY, S. (1980), Sparse complete sets for *NP*: Solution of a conjecture of Berman and Hartmanis *in* "Proceedings 21st IEEE Foundations of Computer Science Symposium," 42–49; *J. Comput. System Sci.* **25** (1982), 130–143.

SEIFERAS, J. (1977), Techniques for separating space complexity classes, *J. Comput. System Sci.* **14**, 73–99.

SEWELSON, V. (1983), "A Study of the Structure of *NP*," Ph. D. thesis, Technical Report 83-575, Department of Computer Science, Cornell University.

STOCKMEYER, L. J. (1976), The polynomial time hierarchy, *Theoret. Comput. Sci.* **3**, 1–22.

WILSON, C. B. (1980), "Relativization, Reducibilities, and the Exponential Hierarchy," Technical Report No. 140/80, Department of Computer Science, University of Toronto, Toronto, Ontario.