On Sparse Sets in NP-P

J. Hartmanis

August 1982

TR 82-508

Department of Computer Science
Cornell University
Ithaca, New York 14853

On Sparse Sets in NP-P

J. Hartmanis

Department of Computer Science

Cornell University

Ithaca, New York 14853

Abstract

The main result of this note shows that there exist sparse sets in *NP* that are not in *P* if and only if *NEXPTIME* differs from *EXPTIME*. Several other results are derived about the complexity of very sparse sets in *NP–P* and an interpretation of the meaning of these results is given in terms of the complexity of solving "individual instances" of problems in *NP–P*.

INTRODUCTION

The work reported in this note was motivated by the desire to understand better what makes the recognition of language in NP hard, provided $P \neq NP$.

It is generally conjectured that $P \neq NP$ [2] and therefore, for example, that it is very hard to determine whether a Boolean formula in conjunctive normal form has a satisfying assignment. 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 whole aggregate of satisfiable Boolean formulas, SAT, but that there are individual instances of formulas for which it is hard to find a satisfying assignment. In particular, we believe that there are syntactically simple, sparse subsets of Boolean formulas for which it cannot be decided in polynomial time whether they are satisfiable.

The results in this note show that the computational difficulty of the SAT recognition problem depends explicitly on what happens to the higher deterministic and nondeterministic time bounded complexity classes. Indeed, the above stated conjecture about difficulty of recognizing sparse subsets of satisfiable formulas is true if and only if the deterministic and nondeterministic exponential time complexity classes do not collapse. More precisely, a *sparse* set is a set that contains up to size n only polynomially many elements. The main result of this note shows:

Theorem: There exists a sparse set in NP–P if and only if

$$EXPTIME \neq NEXPTIME \ .$$

This result can be sharpened to show that the sparse sets in NP–P can be subsets of highly restricted sparse polynomial time recognizable subsets of Boolean formulas.

Corollary: There exists a sparse set S in P such that

$$SAT \cap S \in NP\text{–}P$$

if and only if

$$EXPTIME \neq NEXPTIME .$$

These results have the interesting implication that if $P \neq NP$ and $EXPTIME = NEXPTIME$ then any sparse set in NP is in P. In this case, any syntactical restriction of Boolean formulas to a sparse subset would yield a subclass of formulas for which satisfiability is decidable in polynomial time. Thus only in this case, would the difficulty of deciding satisfiability for Boolean formulas "depend on the existence of the many different types of formulas" and there would be no "difficult individual instances of formulas."

We give several extensions of these results to further emphasize the connection between the computational complexity of subsets of SAT and the behavior of higher complexity classes.

We also relate these problems to the complexity of proving theorems and, since there is ample circumstantial evidence for "individual" instances of theorems that are very hard to prove, we must conjecture that $P \neq NP$ as well as $EXPTIME \neq NEXPTIME$.

It is interesting to note that it has been shown recently by S. Kurtz [4] that there exist oracle sets A such that

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

Clearly, our results imply that for such oracles there are no sparse sets in

$$NP^A - P^A .$$

It should also be recalled that S. Mahaney [6] has shown that if there exists a sparse complete set in NP under many-one polynomial time reductions, then $NP = P$. Even for oracle computations we know from [3] that the existence of a sparse oracle S such that

$$NP \subseteq P^S ,$$

implies that the polynomial hierarchy collapses to Σ_2^P. At the same time we know from Ladner's result [5] that if $P \neq NP$ then incomplete sets exist in $NP-P$.

Combining the above results we know that a sparse set in *NP–P* cannot be complete for *NP* and that furthermore they will exist if and only if *EXPTIME* $\neq$ *NEXPTIME*. If *EXPTIME* = *NEXPTIME* and $P \neq NP$ then incomplete sets exist, but neither the complete nor incomplete sets can be sparse.

SPARSENESS RESULTS

We say that a set S, $S \subseteq \Sigma^*$, is ***sparse*** if and only if there exists a constant k such that

$$|\Sigma^n \cap S| \leq n^k + k .$$

This implies that 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 computations, respectively. Let

$$EXPTIME = \bigcup_{c>1} TIME[2^{cn}] \text{ and } NEXPTIME = \bigcup_{c>1} NTIME[2^{cn}] .$$

Theorem 1: There exists a sparse set S in *NP–P* if and only if

$$EXPTIME \neq NEXPTIME .$$

Proof: If *EXPTIME* $\neq$ *NEXPTIME* then without loss of generality, we can 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:

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

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

$$TALLY(A) \in NP-P .$$

Thus we see that under this assumption there exists a sparse set in *NP–P*.

We now assume that *NEXPTIME* = *EXPTIME* and will show that then every sparse set S in *NP* is already in P.

For the sparse set S let

$$|\Sigma^n \cap S| \leq n^k + k$$

and assume that all the elements of S are lexiographically ordered, $S = \{x_1, x_2, \ldots\}$. Then define

$$S' = \{\#n\ \#i\ \#j\ \#k\ \#d \mid (\exists x, x_1, x_2, \ldots, x_i, y_1, y_2, \ldots, y_j \in S)$$
$$[x_1 < x_2 < \cdots < x_i < x \leq y_1 < y_2 < \cdots < y_j,\ |x| = n, |y_j| \leq n$$
$$\text{and the } k^{th} \text{ digit of } x \text{ is } d\}\ .$$

Since the integers n,i,j,k in the strings in S' are represented in binary, we see that

$$S' \in NEXPTIME\ ,$$

because in this time for input $\#n\ \#i\ \#j\ \#k\ \#d$ the appropriate number of strings of length n or less can be guessed and verified that they are in S and that they satisfy the required conditions. But then, because of our assumption,

$$S' \in EXPTIME\ .$$

From this follows that for input x, $|x| = n$, in polynomial time we can compute the maximal $i + j = m_n$ from the strings

$$\#n\ \#i\ \#j\ \#1\ \#0 \quad \text{or} \quad \#n\ \#i\ \#j\ \#1\ \#1\ ,\ i,j \leq n^k + k$$

(if strings of this type are not in S' then no x of length n is in S). Now each string in S is characterized by a unique i and j such that $i + j = m_n$. Thus x is in S if and only if for some $i_o + j_o = m_n$ the strings

$$\#n\ \#i_o\ \#j_o\ \#k\ \#d \text{ in } S'$$

define x. Since there are only polynomially many possibilities, they can be checked in polynomial time and we see that S is in P, as was to be shown. □

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_o such that all the elements of S, up to size n, can be printed by a deterministic machine in time $n^{k_o} + k_o$. Clearly, every polynomial time

printable set is sparse and it is in P.

Corollary 1: 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 [1]. This guarantees that

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

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

$$g(1^t) \in SAT \text{ then } 1^t \in TALLY(A) \ ,$$

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.

The above result has an interesting interpretation in terms of the difficulty of proving theorems in formal mathematical systems. Let F be an axiomatizable mathematical theory, say Peano Arithmetic or ZF Set Theory. Then, it can easily be seen that the set

T = {Theorem: "Statement of result." Proof: b^k □ | There is a proof of length k or less of the stated theorem in F}

is NP complete. In other words, if we attach to theorems in T the necessary blank tape to write down the formal proof of the stated theorem, then the resulting set is NP complete. Clearly, the original set of theorems is an r.e. complete set. Furthermore, Corollary 1 asserts that there will be a sparse, polynomial time printable set S such that the "sparse subtheory" represented by $S \cap T$ is in $NP\text{-}P$ if and only if $EXPTIME \neq NEXPTIME$.

Since there is a strong conviction that individual theorems in mathematics are hard to prove and that their difficulty does not depend on the existence of the whole aggregate of theorems (in T up to a given length), we must strongly conjecture that $EXPTIME \neq NEXPTIME$, and that there are many "sparse subtheories" which are in $NP\text{-}P$.

The above results can be extended to super sparse sets to emphasize more dramatically the difficulty of proving individual instances of theorems or determining if specific Boolean formulas have satisfying assignments.

We say that a set A, $A \subseteq \Sigma^*$, is *super sparse* if there exists a constant k such that

$$|A \cap \Sigma^n| \leq [\log n]^k .$$

Theorem 2: There exist super sparse sets in $NP-P$ if and only if

$$\bigcup_{r \geq 1} TIME[2^{2^{rn}}] \neq \bigcup_{r \geq 1} NTIME[2^{2^{rn}}] .$$

Proof: Again the proof is easy one way. To prove the implication the other way, assume that a super sparse set S is in NP. Then the set

$$C = \{\#t\ \#i\ \# \mid \text{up to size } 2^t \text{ there exist } i \text{ elements in } S\}$$

is contained in

$$NTIME[2^{2^{rn}}] .$$

Since for x, $|x| \leq 2^t$ the representation of the length of t and i, is bounded by $\log\log|x|$. Therefore in doubly exponential time in the length of the input string $\#t\ \#i\ \#$ we can guess the i strings in S and verify that they are in S. If

$$NTIME[2^{2^{cn}}] \subseteq TIME[2^{2^{dn}}]$$

then we can compute in deterministic double exponential time the maximal i_t of $\#t\ \#i_t\ \#$ in C. This gives the number of strings in S up to size 2^t. But then, a deterministic doubly exponential time machine can compute for input t the sequence of i_t strings in S up to length 2^t,

$$\#x_1\ \#x_2\#\ \cdots\ \#x_{i_t}\ \# .$$

From this it follows that for any x, $|x| = n$, a deterministic polynomial time machine can compute the same string and check if x is in S. Thus if

$$\bigcup_{r \geq 1} NTIME[2^{2^{rn}}] = \bigcup_{r \geq 1} TIME[2^{2^{rn}}]$$

then S is in P. This completes the proof. □

OPEN PROBLEMS

1. We now know that if

$$P \neq NP \quad \text{and} \quad EXPTIME = NEXPTIME$$

then for every sparse set S in P,

$$S \cap SAT \in P \ .$$

At the same time, this does not yet guarantee that for the satisfiable formulas

$$F \text{ in } S \cap SAT$$

we can find satisfying assignments in polynomial time. Clearly, because of the "self reducibility" property of SAT [2], if $SAT \in P$ then for any F in SAT we can find satisfying assignments in polynomial time. Unfortunately, an arbitrary sparse set S in P may not have the self reducibility property for $S \cap SAT$. Thus it still seems to be possible that $P \neq NP$, $EXPTIME = NEXPTIME$ and therefore for all sparse S in P

$$S \cap SAT \text{ in } P \ ,$$

but that there exist sparse sets S' in P such that no polynomial time algorithm can find satisfying assignments for

$$F \text{ in } S' \cap SAT \ .$$

We conjecture that this is the case, but have not been able to prove it so far.

2. One would expect that the existence of sets with density slightly above polynomial in NP–P would guarantee that the deterministic and nondeterministic time classes slightly below exponential time would not collapse. For example, we conjecture that there exist $n^{c \cdot \log n}$-dense sets in NP–P if and only if

$$\bigcup_{c \geq 1} TIME[2^{c\sqrt{n}}] \neq \bigcup_{c \geq 1} NTIME[2^{c\sqrt{n}}] \ .$$

Unfortunately, so far we have not been able to prove this conjecture and to extend these results

to sets with higher densities.

3. We also conjecture that there exist sparse sets in $NLOGTAPE - LOGTAPE$ if and only if the sets of deterministic and nondeterministic context-free languages are not equal, i.e.

$$DCSL \neq NDCSL \ .$$

Unfortunately, the techniques used in the proof of Theorem 1 do not apply directly and so far we have not been able to verify this conjecture. The main difficulty stems from the fact that on $NLOGTAPE$ we do not have enough memory to guess and count the possible strings of the sparse sets, as it was done for NP.

References

[1] Berman, L., and J. Hartmanis, "On Isomorphisms and Density of *NP* and Other Complete Sets", *SIAM Journal of Computing 6*:2 (June 1977), 305-322.

[2] Garey, M.R., and D.S. Johnson, *Computers and Intractability, A Guide to the Theory of NP-Completeness*, W.H. Freeman and Co., San Francisco, California, 1979.

[3] Karp, R.M., and R.J. Lipton, "Some Connections Between Nonuniform and Uniform Complexity Classes", *Proceedings 12th Annual ACM Symposium on Theory of Computation* (April 1980), 302-309.

[4] S. Kurtz, Private communication.

[5] Ladner, R.E., "On the Structure of Polynomial Time Reducibility", *Journal of the ACM* (1975), 155-171.

[6] Mahaney, S., "Sparse Complete Sets for NP: Solution of a Conjecture of Berman and Hartmanis", *Proceedings 21st IEEE Foundations of Computer Science Symposium* (1980), 42-49.