Limitations of the Upward Separation Technique*

Eric Allender[†]
Department of Computer Science
Rutgers University
New Brunswick, N.J. 08903

April, 1990

^{*}A preliminary version of this paper was presented at the 16th International Colloquium on Automata, Languages, and Programming [3].

[†]Supported in part by National Science Foundation Research Initiation Grant number CCR-8810467.

SUMMARY

The upward separation technique was developed by Hartmanis, who used it to show that E=NE iff there is no sparse set in NP-P [15]. This paper shows some inherent limitations of the technique. The main result of this paper is the construction of an oracle relative to which there are extremely sparse sets in NP-P, but NEE = EE; this is in contradiction to a result claimed in [14, 16]. Thus, although the upward separation technique is useful in relating the existence of sets of polynomial (and greater) density in NP-P to the NTIME(T(n)) = DTIME(T(n)) problem, the existence of sets of very low density in NP-P can not be shown to have any bearing on on this problem using proof techniques that relativize.

The oracle construction is also of interest since it is the first example of an oracle relative to which EE = NEE and $E \neq NE$. (The techniques of [10], [17], [20], and [24] do not suffice to construct such an oracle.) The construction is novel and the techniques may be useful in other settings.

In addition, this paper also presents a number of new applications of the upward separation technique, including some new generalizations of the original result of [15].

1 Introduction

It has long been known that $E \neq NE$ iff there is a tally set in NP - P [9]; thus $E \neq NE$ $\Longrightarrow P \neq NP$. This is an example of downward separation. Motivated by the question of whether or not the converse implication could be proved, Dekhtyar [12] and Wilson [29] constructed oracles relative to which $P \neq NP$ and E = NE; thus the P = NP question cannot be shown to be equivalent to the E = NE question by any relativizable proof technique.

Note that Wilson's construction gives rise to an oracle relative to which $P \neq NP$, but NP-P contains no tally sets. In [20], Kurtz presented a seemingly stronger oracle construction, relative to which $P \neq NP$ and NP-P contains no *sparse* sets. However, shortly after Kurtz's results had been announced, Hartmanis showed that the two oracle constructions were equivalent, in the sense that relative to any oracle, there is a tally set in NP-P iff there is a sparse set in NP-P [15]. The proof technique used to prove this result was dubbed the *upward separation technique*.

The upward separation technique of [15] was generalized and extended in [16] in a number of ways. One generalization, considered in [15] and [16], relates the existence of very sparse sets in NP-P to the separation of the classes NTIME(T(n)) and DTIME(T(n)) for certain large functions T. Let NEE denote the class NTIME $(2^{O(2^n)})$ and let EE denote DTIME $(2^{O(2^n)})$. It is claimed in [15, 16] that EE=NEE iff there are no supersparse sets in NP-P; however we show that this result cannot be obtained using

the upward separation technique, or by using any relativizable proof technique.¹ We present an oracle relative to which EE=NEE and NP-P contains extremely sparse sets.

The technique used to construct this oracle is different from that used in earlier constructions of this sort. It seems as though these techniques may be applicable to a variety of other problems. The following few paragraphs explain some of what makes our methods novel.

As a corollary of the main theorem of this paper, there is an oracle relative to which EE=NEE and $E\neq NE$. A number of other authors have constructed oracles relative to which, for example, E=NE and $P\neq NP$, and it would seem that this question might yield to similar techniques, but that turns out not to be the case.

Since Dekhtyar's paper [12] gives no proofs, we cannot speak of the limitations of his methods, however the techniques used by Kurtz [20], Wilson [29, 10] and in the later related work of Lischke [23, 24] all rely on interleaving alternating stages of encoding information into the oracle (e.g., to ensure that E=NE) and diagonalizing (e.g., to ensure that $P\neq NP$). In order for the technique to work, it is necessary that "interference" between the stages be avoided.

However, it seems that the techniques used by the authors mentioned above cannot successfully avoid interference between stages when one is trying to diagonalize to keep

¹The proofs of Theorem 2.4 in [15] and Theorem 6 in [16] essentially show that, if EE=NEE, then the census function for any supersparse set L in NP can be computed in polynomial time. However, they then incorrectly conclude that it is possible to actually construct the elements of L.

 $E\neq NE$ while doing encoding to make some larger complexity classes coincide. (That is, the combinatorial arguments, which guarantee that interference between stages can be avoided, break down when one is trying to guarantee $E\neq NE$.) Thus, for example, no oracle had been constructed relative to which EE=NEE and $E\neq NE$.

Another approach was used by Heller [17]. Using alternating stages of encoding, Heller constructed an oracle relative to which $E=NE \subseteq \Delta_2^p$. Note that $P \neq NP$ relative to his oracle, since P=NP implies $\Delta_2^p = P$. This technique cannot be used to build an oracle relative to which EE=NEE and $E\neq NE$, since it was shown by [16] that, in relativized computations, E=NE does not imply that the exponential-time hierarchy collapses.

In the oracle construction of this paper, the "encoding" is done at the start, rather than in a series of stages. The "diagonalization" is carried out, in essence, by diagonalizing against all possible information that could be provided by the encoding phase. The technique seems to be powerful and applicable to other situations. A number of generalizations of this technique are worked out in [5].

It is also worth mentioning that the oracle construction makes use of Kolmogorov complexity as a tool for simplifying the combinatorial arguments. Hartmanis [14] was the first to use Kolmogorov complexity as a tool in oracle constructions; a number of applications of this technique are presented in [13].

Although the main result of this paper is a demonstration that certain generalizations of the upward separation results do not hold, it is useful to examine some generalizations

that do hold. Some of these are presented in the final section of this paper. Among other things, we show that:

There is a log-sparse set in FewP-P iff FewE \neq E. There is a O(1)-sparse set in UP-P iff UE \neq E.

(Terms such as "log-sparse" and "O(1)-sparse" are defined in Section 2.) We also show that the following are equivalent:

- 1. There is a subset of $K[O(\log \log n), n^{O(1)}]$ in NP-P.
- 2. There is a subset of $K[O(\log n), n^{O(1)}]$ in NP-P with polylogarithmic density.
- 3. For some k, DTIME $(2^{O(2^{n/k})}) \neq \text{NTIME}(2^{O(2^{n/k})})$.

2 Preliminaries

It is expected that the reader will be familiar with basic concepts from complexity theory, such as Turing machines, circuits, and complexity classes such as P, NP, etc. For background and definitions, see e.g. [18], [27], [6]. We will use E and NE to refer to $DTIME(2^{O(n)})$ and $NTIME(2^{O(n)})$, respectively. EE and NEE denote $DTIME(2^{O(2^n)})$ and $NTIME(2^{O(2^n)})$, respectively.

For any string x, the length of x is denoted by |x|. For any set S, |S| denotes the cardinality of S. All languages considered in this paper are subsets of $\{0,1\}^*$. We will as-

sume a standard mapping from $\{0,1\}^*$ onto the positive integers; namely the string x will denote the integer whose binary representation is 1x. Thus, for example, $|x| = \lfloor \log x \rfloor$, and given strings x and y, we may say $x \leq y$ (which corresponds to the lexicographic ordering on $\{0,1\}^*$). We will use strings and the integers they represent interchangeably, as for example, in the expression 2^{x10} , where x is a string.

Given any sets A and B, the set $\{1x : x \in A\} \cup \{0x : x \in B\}$ is denoted by A \oplus B.

Given any language L, the census function for L, denoted $c_L(n)$, is defined to be the number of strings in L of length at most n. We may sometimes say that L has census f(n) (or density f(n)) if $c_L(n) \leq f(n)$. A set L is said to be sparse if $c_L(n) = n^{O(1)}$. L is said to be supersparse if $c_L(n) = O(\log n)$. We say that L is log sparse if L contains $O(\log n)$ strings of length n, and L is O(1)-sparse if L contains at most k strings of each length n, for some k independent of n. Note that the class of supersparse sets is incomparable with the class of O(1)-sparse sets.

Kolmogorov complexity and generalized Kolmogorov complexity provide a framework for talking about the complexity of individual strings. The definitions and facts relating to Kolmogorov complexity that are used in this paper are standard and elementary; background information can be obtained, for example, from the introduction to [25]. More extensive background and an excellent bibliography can be found in [22]. We will let K(x) denote the Kolmogorov complexity of x (that is, we will fix some universal Turing machine M, and K(x) will be the length of the smallest y such that M, on input y, outputs x).

Here are some simple notions concerning Kolmogorov complexity that will be useful in the proof of Theorem 1. For any string $y = y_1 y_2 \dots y_n$, let \overline{y} denote the "self-delimiting" version of y: $y_1 0 y_2 0 \dots y_{n-1} 0 y_n 1$. Note that $|\overline{y}| = 2|y|$. Self-delimiting strings are a useful tool when proving easy upper bounds on the Kolmogorov complexity of strings. In particular, they are useful when supplying several arguments as input to a Turing machine. The next two paragraphs illustrate the use of self-delimiting strings.

It is easy to show that there are strings w and z such that K(wz) < K(z). (For example, consider the case when $wz = 0^{2^{2^k}}$.) On the other hand, it is intuitively clear that wz can't have complexity much smaller than z, and the following (standard) argument makes this precise.

Let y be a description of wz (i.e., let M(y) = wz), let n be the length of z, and let v be the description of a Turing machine that, on input $\overline{n}y$, outputs the last n bits of M(y). Note that $M(\overline{v}\,\overline{n}y = z)$. This shows that, for all w and z, $K(z) \leq K(wz) + 2|z| + 2|v|$.

Generalized Kolmogorov complexity takes into account the time complexity of computing a string from its description. The definitions we use were introduced in [14]: Given any Turing machine M_v , we define $K_v[s(n), t(n)]$ to be the set of all strings x such that, for some string y of length at most s(|x|), M_v prints out x on input y in at most t(|x|) steps. As was shown in [14], there is a machine M_u (a universal Turing machine) such that, for all v there exists a constant v (depending on v) such that $K_v[s(n), t(n)] \subseteq K_u[s(n) + c, ct(n) \log t(n) + c]$. Dropping the subscript, we will choose some particular universal Turing machine M_u and let K[s(n), t(n)] denote $K_u[s(n), t(n)]$.

3 Construction of the Oracle

Theorem 1 Let $d: \mathbb{N} \to \mathbb{N}$ be any nondecreasing unbounded function. Then there is an oracle B such that $NEE^B = EE^B$ and there is a set $L \in \mathbb{NP}^B - \mathbb{P}^B$ with census $c_L(n) < d(n)$.

Proof: First, let us establish some notation. Let N_1, N_2, \ldots be an enumeration of nondeterministic Turing machines, and let P_1, P_2, \ldots be an enumeration of deterministic polynomial-time Turing machines, such that P_j runs in time n^j .

Let s be a slow-growing surjective function computable in polynomial time. (Choosing $s(n) = \log^* n$ will suffice.) Let $r(n) = \min\{m : s(m) = n\}$; note that r is essentially s^{-1} .

For any set $A \subseteq \Sigma^*$, define

$$Q(\mathbf{A}) = \{0^{2^x} 10^{r(j)} : N_j^{\mathbf{A} \oplus Q(\mathbf{A})} \text{ accepts } x \text{ in } \le 2^{2^{|x|}} \text{ steps}\}$$

Note that Q(A) is well-defined, since for any y of the form $0^{2^x}10^{r(j)}$ it follows that $|y| > 2^{2^{|x|}}$, and thus N_j cannot query y in $2^{2^{|x|}}$ steps.

We claim that for every set A, $\mathrm{EE}^{\mathrm{A}\oplus Q(\mathrm{A})} = \mathrm{NEE}^{\mathrm{A}\oplus Q(\mathrm{A})}$. To see this, let L \in $\mathrm{NEE}^{\mathrm{A}\oplus Q(\mathrm{A})}$, so L is accepted by some machine $N_i^{\mathrm{A}\oplus Q(\mathrm{A})}$ in time $2^{2^{n+c}}$ for some c. Let N_j be a machine that, on input $x0^c$, simulates N_i on input x. N_j runs in time 2^{2^n} , and $x \in$ L $\iff 0^{2^{x0^c}}10^{r(j)} \in Q(\mathrm{A})$. This query can be asked in time $2^{O(2^n)}$.

Let
$$L(A) = \{x : \exists y \ (xy \in A \text{ and } |x| = |y|)\}$$
. For all sets $A, L(A) \in NP^A \subseteq$

 $NP^{A\oplus Q(A)}$. It will thus suffice to build a set A such that $c_{L(A)}(n) \leq d(n)$ and such that, for all j, the set accepted by P_j with oracle $A \oplus Q(A)$ differs from L(A).

Let c be a large constant. At a certain point in the proof, we will use the fact that c is larger than the description of some simple Turing machines. Choosing $c = 10^6$ should be sufficient, depending on the particular encoding conventions.

The construction is done by the initial segment method. The (finite) oracle constructed by the end of stage j will be denoted by A_j . To begin the construction, let $A_0 = \emptyset$.

To construct A_j $(j \ge 1)$, first choose $n = n_j$ so that:

- 1. n is even
- $2. (n_{j-1})^{j-1} < n$
- 3. $|L(A_{j-1})| + 1 < d(n)$
- 4. $c + 2\log j + 2(j-1)n_{j-1} + 2s(n^j)\log(n^j) + 2\log n^j + n/2 < n c$.

(The right hand side of the fourth inequality is n/2 + o(n); thus note that all large even n satisfy all of these criteria.)

Let $Q = \{z : |z| \le n^j \text{ and } z \text{ is of the form } 0^{2^y} 10^{r(i)} \text{ for some } y \text{ and } i\}$. Note that $|Q| \le s(n^j) \log(n^j) < n/2$. When P_j computes on inputs of length n, all queries to the Q(A) part of the oracle will be answered negatively for queries not in Q. That is, knowing

membership in Q(A) for all elements of Q is sufficient to enable us to answer all queries to the Q(A) part of the oracle.

An element $q \in \{0,1\}^{|Q|}$ will be called a Q-vector. We will write $P_j^{A \oplus q}(x)$ to denote the outcome of the computation of P_j on input x, where all queries to the Q(A) part of the oracle are answered according to the vector q. That is, if P_j asks the Q(A) part of its oracle about a string not in Q, the oracle responds negatively, and if P_j asks about the i-th element of Q, then the oracle responds with the i-th bit of q.

Let W =
$$\{w0^{n/2} : |w| = n/2\}.$$

Given any Q-vector q, we associate with it a set $F(q) \subseteq W$ as follows:

$$F(q) = \{ x \in W : P_j^{A_{j-1} \oplus q}(x) = 1 \}.$$

Since there are only $2^{|Q|} < 2^{n/2} = |W|$ Q-vectors, there must be some $w \in W$ such that for all $q \in Q$, $F(q) \neq \{w\}$. Choose one such w, and let z be a string of length n of maximal Kolmogorov complexity (i.e. $K(z) \geq n$).

Let $A_j = A_{j-1} \cup \{wz\}$, and let q be the Q-vector encoding $Q(A_j)$.

Note that $W \cap L(A_j) = \{w\} \neq F(q) = W \cap \{x : P_j^{A_{j-1} \oplus q}(x) = 1\}$. Thus in order to show that P_j with oracle $A_j \oplus Q(A_j)$ does not accept $L(A_j)$, it will suffice to show that for all $x \in W$, $P_j^{A_{j-1} \oplus q}(x) = P_j^{A_j \oplus q}(x)$. If this can be shown, then for all $x \in W$, x is accepted by P_j with oracle $A_j \oplus Q(A_j) \iff x \in F(q)$, and thus the set accepted by P_j

with oracle $A_j \oplus Q(A_j)$ differs from $L(A_j)$ on some element of W.

Claim:
$$\forall x \in W, P_j^{\mathbf{A}_{j-1} \oplus q}(x) = P_j^{\mathbf{A}_j \oplus q}(x).$$

<u>Proof of claim:</u> Since $A_{j-1} \oplus q$ and $A_j \oplus q$ differ only on the string wz, the outcome of $P_j^{A_{j-1} \oplus q}(x)$ and $P_j^{A_j \oplus q}(x)$ can only differ if wz is queried on each of these computations. We will show that it cannot be queried.

Note that, since z is simply the last half of wz, $K(z) \leq K(wz) + c$, so $K(wz) \geq n - c$.

Note also that any query asked during the computation of $P_j^{\Lambda_{j-1}\oplus q}(x)$ can be reconstructed from

- *j*,
- a description of A_{j-1} (which need be no longer than $2(j-1)n_{j-1}$, since A_{j-1} consists of exactly j-1 strings, each of length $\leq n_{j-1}$),
- q (which has length $s(n^j)\log(n^j)$),
- a number between 1 and n^{j} , indicating the step at which the query is made,
- a description of x (which need be no longer than n/2 + c, since $x \in W$ and thus is of the form $y0^{n/2}$).

We can now conclude that the Kolmogorov complexity of any query is at most $c + 2\log j + 2(j-1)n_{j-1} + 2s(n^j)\log(n^j) + 2\log n^j + n/2 < K(wz)$. Thus wz is not queried.

We should remark that the same techniques suffice to prove much stronger results. For example, a slight modification of this proof will show that there is an oracle relative to which $DTIME(2^{O(n \log n)}) = NTIME(2^{O(n \log n)})$ and there are extremely sparse sets in NP-P. (More generally, if T is any suitably nice time bound, such that $T(\log n)$ grows more quickly than any polynomial, then there is an oracle relative to which there are very sparse sets in NP-P and NTIME(T(n+O(1))) = DTIME(T(n+O(1))). Interested readers may wish to consult [5], where a similar generalization is proved.) Recall that $E\neq NE$ relative to all of these oracles, since relative to any oracle, the existence of sparse sets in NP-P implies $E\neq NE$, since the upward separation technique relativizes.

4 New Upward Separation Results

Although the results of the last section tell us that we cannot hope to relate the existence of very sparse sets in NP-P to the question of whether or not nondeterministic and deterministic time coincide on large time bounds, we can nevertheless salvage some useful generalizations of the upward separation technique. This section presents those generalizations.

The results of this section can be briefly summarized as follows:

The upward separation technique can be used to

• relate the existence of sets of low density and moderate Kolmogorov complexity in NP-P to the existence of sets of very low Kolmogorov complexity in NP-P.

- relate the DTIME(T(n)) vs NTIME(T(n)) question to the existence of sets of low density and moderate Kolmogorov complexity in NP-P (for T in a certain range).
- relate the DTIME(T(n)) vs NTIME(T(n)) question to the existence of sets of polynomial and greater density in certain smaller complexity classes.
- relate the FewE=E and UE=E questions to the existence of certain types of sparse sets in FewP-P and UP-P.

Recall that the upward separation results tell us that there is a sparse set in NP-P iff there is a tally set in NP-P. Equivalently, if there is any sparse set in NP-P, then there is a set of low generalized Kolmogorov complexity in NP-P. Theorem 3, given below, is a generalization of this result, for sets of very low density.

In order to express the result in its full generality, it is useful to define a somewhat more restricted notion of generalized Kolmogorov complexity.

Definition: For any Turing machine M_v , and any functions s and t, define the set $\mathrm{KU}_v[s(n), t(n)]$ to be the set of all strings x such that there is exactly one string y such that $|y| \leq s(|x|)$ and M_v , on input y, prints out x in at most t(|x|) steps.

It is appropriate to make a few observations about the sets of the form $KU_v[s(n), t(n)]$. It was shown in Theorem 9 of [2] that it is impossible to define a "universal" Turing machine that would allow one to suppress the subscript v. Some other relevant facts are given in the following proposition.

Proposition 2

- For any function $s(n) = O(\log n)$, and for any k, there is a machine v and a constant l such that $K[s(n), n^k]$ is contained in $KU_v[s(n), n^l]$.
- There is a machine M_v such that $\mathrm{KU}_v[n,n] = \Sigma^*$ (= $\mathrm{K}[2n,n^k]$ for any k > 1).

Proof: Part 2 is obvious. To see part 1, let s and k be given, and note that there is some a such that $\forall n \ s(n) \le a \log n + a$. Let M_v operate as follows: on input y, run M_u on input y, and let $x = M_u(y)$; let n = |x|. For all z of length at most $a \log n$, run $M_u(z)$ for n^k steps. If no z < y is found such that $M_u(z) = x$, then output x; else output the yth string in the lexicographic ordering that is not equal to $M_u(z)$ for any of those z.

It is clear that if x has a short description on machine M_u , then it has an equally short description on M_v . Also, no string of length n has two descriptions of length $\leq a \log n$.

If s(n) is superlogarithmic but s(n) < n, then no nontrivial upper bound is known for the KU complexity of sets of the form $K[s(n), n^k]$. That is, for example, it is not known if there is some v and k such that $K[\log^2 n, n^2] \subseteq KU_v[n/2, n^k]$. For more information concerning these questions, see [2].²

Theorem 3 Let $s(n) = \Omega(\log \log \log n)$, and $s(n) = O(\log n)$. Then the following are equivalent:

²Unfortunately, some of the information presented in [2] concerning the sets $KU_v[s(n), n^k]$ is erroneous. It is stated in [2] that, for all v and k, $KU_v[s(n), n^k]$ is in UP, whereas in fact all that is known is that this set is in US, where US is the class corresponding to "Unique Satisfiability", as studied in [8].

- 1. $\exists k \; \exists v \; \text{NP-P contains a subset of } \mathrm{KU}_v[ks(n), n^k].$
- 2. $\exists k \text{ NP-P contains a subset of } K[ks(n), n^k].$
- 3. $\exists k \; \exists v \; \text{NP-P} \text{ contains a set with census } 2^{ks(n)} \text{ that is a subset of } \mathrm{KU}_v[2^{ks(n)}, n^k].$

Proof:

- $((1) \implies (3))$ is immediate (with no restriction on s), and $((2) \implies (1))$ follows from Proposition 2 (requiring only that $s(n) = O(\log n)$).
 - $((3) \implies (2))$: This is where the upward separation technique is used.

Let L be a set in NP – P that has census $2^{ks(n)}$, such that L \subseteq KU_v[$2^{ks(n)}$, n^k]. Define L' to be the set:

 $\{1^n0j: \exists i\ n=2^{2^i} \text{ and there are at least } j \text{ strings in L of length at most } n\} \cup \{0^n1\langle j,h,l,b\rangle: \exists i\ n=2^{2^i} \text{ and there is a sorted list of } j \text{ strings in L,}$ $x_1,x_2,\ldots,x_j, \text{ each of length at most } n, \text{ and there is a string } y \text{ of length}$ at most $2^{ks(|x_h|)}$ such that, on input y, M_v outputs x_h , and the time required by $M_v(y)$ is at most $|x_h|^k$, and the lth bit of y is b.

It is obvious that L' is in NP. Furthermore, since any string of length n in L' can be described by a tuple of the form $\langle i, j \rangle$ or $\langle i, j, h, l \rangle$, where each of i, j, h, and l have binary representations of length at most ks(n), it is clear that L' is a subset of $K[k's(n), n^2]$ for some k'.

It remains only to show that L' is not in P. Assume that L' is in P; it will follow that L is in P, contrary to our assumptions. On input x of length n, the following polynomial-time algorithm will determine if $x \in L$.

Find i so that $n \leq 2^{2^i} \leq n^2$.

Let m denote 2^{2^i} .

Find the largest j such that 1^m0j is in L'. (Note: $j \leq 2^{ks(m)}$.)

For h = 1 to j:

For l = 1, 2, ... and $b \in \{0, 1\}$, determine if $0^m 1\langle j, h, l, b \rangle$ is in L'.

Use this information to build a description y, such that $M_v(y)$ is the hth element in L.

(Note that the fact that there is a *unique* description is crucial here.)

If x is one of the j elements of L found in this way, accept; else reject.

If more restrictions are placed upon s, then the results of Theorem 3 can be stated without mentioning sets of the form $KU_v[s(n), t(n)]$.

Corollary 4 Let $s(n) = \Omega(\log \log \log n)$, and $s(n) = o(\log \log n)$. Then the following are equivalent:

- 1. $\exists k \text{ NP-P contains a subset of } K[ks(n), n^k].$
- 2. $\exists k \text{ NP-P}$ contains a set with census $2^{ks(n)}$ that is a subset of $K[2^{ks(n)}, n^k]$.

Proof: Follows from Theorem 3, using the result of part 1 of Proposition 2, which is

applicable since $2^{ks(n)} = O(\log n)$.

Of course, part of the appeal in the original upward separation results of [14, 16] lies in the relationships that were drawn between the NTIME(T(n)) = DTIME(T(n)) question for large T and the existence of certain kinds of sets in NP-P. We were only able to draw these relationships for T in the range between 2^n and 2^{2^n} , as presented in the following theorem.³

Theorem 5 Let T(n) be a nondecreasing time-constructible function such that $T(n) = 2^{O(2^n)}$ and $T(n) = 2^{\Omega(n)}$. Let d(m) = 0 the greatest r such that $T(r) \leq m$. Then the following are equivalent:

- 1. $\exists k \text{ NTIME}(T(\lceil n/k \rceil)^{O(1)}) \neq \text{DTIME}(T(\lceil n/k \rceil)^{O(1)}).$
- 2. $\exists k \; \exists v \; \text{NP-P} \text{ contains a subset of } \mathrm{KU}_v[kd(n), n^k].$
- 3. $\exists k \text{ NP-P contains a subset of } K[kd(n), n^k].$
- 4. $\exists k \ \exists v \ \text{NP-P}$ contains a set with census $2^{kd(n)}$ that is a subset of $\mathrm{KU}_v[2^{kd(n)}, n^k]$.

Proof: By Theorem 3, (2), (3) and (4) are all equivalent. It will suffice to show (1) \iff (3).

((1) \Longrightarrow (3)): Let L be in NTIME $(T(\lceil n/k \rceil)^{O(1)})$ – DTIME $(T(\lceil n/k \rceil)^{O(1)})$. Let A = $\{x10^{T(\lceil |x|/k \rceil)} : x \in L\}$. It is immediate that L \in NP – P. (This is a standard translational

³Regrettably, [3] claimed the results of Theorem 5 for T satisfying $T(n) = 2^{\Omega(n)}$ and $T(n) = O(2^{2^{2^n}})$. It is not known if Theorem 5 holds for T in the high end of this range.

argument; see, e.g., [9].) It is also clear that L is contained in K[kd(n), 2n]. (Note that for all s, $d(T(s)) \ge s$, and thus if $n = |x10^{T(m)}|$, then $|x| \le km \le kd(T(m)) \le kd(n)$.)

 $((3) \Longrightarrow (1))$: In order to simplify the proof of this direction, assume that our "universal" Turing machine M_u has the property that for all y and z, if $M_u(y)$ is defined, then $M_u(y) = M_u(yz)$. (This is a standard "self-delimiting" assumption about Kolmogorov complexity; a discussion of this notion can be found in section 2.7 of [22], among other places.) This assumption has no effect on any of the other results in this paper.

Let L be a subset of $K[kd(n), n^k]$ in NP – P. Let L' be the set $\{y : \text{ on input } y, M_u \text{ outputs a string } x \text{ of length at most } T(\lceil |y|/k \rceil) \text{ in time } |x|^k, \text{ and } x \text{ is in L}\}$. It is immediate that L' is in NTIME $(T(\lceil n/k \rceil)^{O(1)})$.

If L' is in DTIME $(T(\lceil n/k \rceil)^{O(1)})$, then the following algorithm shows that L is in P, contrary to our assumptions on L.

Since $T(n) = 2^{\Omega(n)}$, there is some constant a such that $d(n) \leq a \log n$. The algorithm below makes use of this constant.

On input x of length n,

Find the shortest string y of length at most $a \log n$ such that $M_u(y) = x$ in time n^k , if such a string y exists. (If not, reject x.)

If $T(\lceil |y|/k \rceil) > n$, then reject (since $x \notin K[kd(n), n^k]$).

Otherwise, let r be the least integer such that $T(\lceil (|y|+r)/k \rceil) \ge n$. (Since T is at most doubly-exponential, it follows that $T(\lceil (|y|+r)/k \rceil)$ is bounded by a polynomial in n.)

Accept x iff $y1^r \in L'$.

For the special case when T(n) is doubly-exponential, we can rephrase the results of Theorem 5 without making mention of sets of the form $KU_v[s(n), t(n)]$.

Corollary 6 The following are equivalent:

- 1. $\forall l \text{ NTIME}(2^{O(2^{n/l})}) = \text{DTIME}(2^{O(2^{n/l})}).$
- 2. For all k, every subset of $K[k \log \log n, n^k]$ in NP is in P.
- 3. For all k, every subset of $K[k \log n, n^k]$ of polylogarithmic density in NP is in P.

Note that in going from item (2) to item (3) of this corollary, we do not quite achieve an exponential jump in the size of the allowable descriptions. That is, in item (3), we only speak of $K[k \log n, n^k]$, instead of $K[\log^k n, n^k]$. It is not known if the existence of a subset of $K[\log^k n, n^k]$ of polylogarithmic density in NP-P implies the existence of a subset of $K[l \log n, n^l]$ in NP-P for some l.

Of course, the original result of [14, 16] also follows from Theorem 5, when $T(n) = 2^n$. In this case, d(n) = n, and thus there is no reason to mention Kolmogorov complexity at all.

Corollary 7 [14, 16] The following are equivalent:

 \bullet E=NE

• Every sparse set in NP is in P.

Theorem 5 can be generalized to take into account functions T(n) that are subexponential or more than doubly exponential. The first of these generalizations may be found in [16], where it is shown that there are no $n^{\log n}$ "uniformly distributed dense" sets in NP-P iff NTIME($2^{O(\sqrt{n})}$) = DTIME($2^{O(\sqrt{n})}$). (The definition of "uniformly distributed density" can be found in [16].) Similarly, if we wish to generalize Theorem 5 to take into account rapidly-growing functions T(n), it seems that we need to talk about sparse sets containing only strings of easily-described lengths (e.g., lengths n, where n is a string of low generalized Kolmogorov complexity, or lengths n that are in the range of T); that is, the sparse sets need to have their strings "nicely distributed".

Thus far in this section, we have discussed generalizations of the upward separation technique, relating the NTIME vs. DTIME question to the existence of very sparse sets in complexity classes. The upward separation technique also enables us to relate the NTIME vs. DTIME question to the existence of sets of more than polynomial density in certain complexity classes, as shown by the following theorem (which generalizes Corollary 5 in [16]).

Theorem 8 Let T be a time-constructible function, $T(n) \ge n$, and let d(n) be a nondecreasing function such that (1) $n \le d(n) \le 2^n$, (2) $d(n+1) = d(n)^{O(1)}$, and (3) $\log d(n)$ is computable in time $T(d(n)^c)$ for some c. Then

$$DTIME(T(2^{O(n)})) = NTIME(T(2^{O(n)}))$$

iff there is no set of census $\leq d(n)$ in $\text{NTIME}(T(d(n)^{O(1)})) - \text{DTIME}(T(d(n)^{O(1)}))$.

Proof: (\Longrightarrow) Let L be a set with census $\leq d(n)$ in NTIME $(T(d(n)^k))$. Define L' to be the set

 $\{1\langle m,d(m),j\rangle:$ there are at least j strings in L of length $m\}\cup$ $\{0\langle m,d(m),j,h,l,b\rangle:$ there is a sorted list of j strings in L, $x_1,x_2,\ldots,x_j,$ each of length m, and the lth bit of x_h is $b\}.$

L' is in NTIME $(T(2^{O(n)}))$, since (a) in time approximately $mj=2^{O(|1\langle m,d(m),j\rangle|)}$ a nondeterministic machine can guess j strings of length m, and (b) on input $1\langle m,d(m),j\rangle$ of length n, a nondeterministic machine can check that a string of length m is in L in time $T(d(m)^k) \leq T((2^n)^k)$.

By assumption, then, L' is in DTIME $(T(2^{O(n)}))$. Thus the following deterministic algorithm for recognizing L will run in time $T(d(n)^{O(1)})$:

On input x of length n,

Compute d(n).

Using fewer than d(n) calls to L', find the largest j such that $1\langle n, d(n), j \rangle \in L'$.

For
$$h \in \{1, ..., j\}$$
, $l \in \{1, ..., n\}$, and $b \in \{0, 1\}$
determine if $0\langle n, d(n), j, h, l, b \rangle \in L'$.

Use this information to list all j elements of length n in L.

If x appears in this list, accept; else reject.

(\iff) Assume that every set with census bounded by d(n) in NTIME $(T(d(n)^{O(1)}))$ is in DTIME $(T(d(n)^{O(1)}))$, and let L be in NTIME $(T(2^{O(n)}))$. Then the set L' = $\{x10^m: d(m+|x|+1) \geq 2^{|x|} \text{ and } x \in L\}$ is a set of census $\leq d(n)$ in NTIME $(T(d(n)^{O(1)}))$. By assumption, L' is in DTIME $(T(d(n)^{O(1)}))$.

Here is a deterministic algorithm for recognizing L that runs in time $T(2^{O(n)})$: On input x of length n,

Find the least m such that $d(n+m+1) \geq 2^n$.

(By the conditions on d, $d(n+m+1)=2^{O(n)}$.)

Accept iff $x10^m \in L'$.

Corollary 9 Let T be any time-constructible function. Then

$$\operatorname{NTIME}(T(2^{O(n)})) = \operatorname{DTIME}(T(2^{O(n)}))$$

iff there is no sparse set in $NTIME(T(n^{O(1)})) - DTIME(T(n^{O(1)}))$.

Corollary 10 DTIME $(2^{2^{O(n)}}) = \text{NTIME}(2^{2^{O(n)}})$ iff there is no sparse set in NTIME $(2^{n^{O(1)}})$ – DTIME $(2^{n^{O(1)}})$.

It is not known if the EE=NEE question is in fact equivalent to any questions about sparse or supersparse sets. However, Corollaries 10 and 6 relate somewhat to EE and NEE.

Note that the results of Section 3 show that Theorem 8 can not be generalized to take

care of the situation where the density d(n) is sublinear, without taking Kolmogorov complexity into account.

Finally, let us present some new applications of the upward separation technique. In [1], the class FewP was introduced (see also [4]; the class was called FNP in [1]). FewP is the class of all sets accepted by NP machines such that, if input x is accepted, the number of accepting paths is polynomial in the length of x. FewP has also been studied by a number of other authors (e.g. [7, 11, 21, 26]). One of the main results in [4] and [1] was

Theorem 11 [4] Every sparse set in P is P-printable iff there are no sparse sets in FewP-P.

Since the upward separation technique is useful in showing that questions about sparse sets in NP are equivalent to questions about NE, it was natural to wonder if the statement "there are no sparse sets in FewP-P" is in fact equivalent to some relation between exponential-time complexity classes. Thus the class FewE was defined in [1]; FewE is the class of languages accepted by NE machines such that, if input x of length n is accepted, there are at most $2^{O(n)}$ accepting paths. It is easy to show that FewE=E iff there are no tally sets in FewP-P. This leads naturally to the question of whether the upward separation technique can show that FewE=E iff there are no sparse sets in FewP-P.

We are unable to answer that question, but we can use upward separation to show

the following result:

Theorem 12 FewE=E iff there are no log-sparse sets in FewP-P.

Proof: (\iff) This direction follows from the easy observation that FewE=E iff there are no tally sets in FewP-P.

 (\Longrightarrow) Assume that FewE = E, and let L be a log-sparse set in FewP. Let L' be the set

 $\{1\langle m,j\rangle: \text{ there are at least } j \text{ strings in L of length } m\} \cup \{0\langle m,j,h,l,b\rangle: \text{ there is a sorted list of } j \text{ strings in L}, x_1,x_2,\ldots,x_j, \text{ each of length } m, \text{ and the } l\text{th bit of } x_h \text{ is } b\}.$

It is easy to observe that there is an NE machine accepting L' with the property that, on inputs of the form $1\langle m, j \rangle$ or $1\langle m, j, h, l, b \rangle$, of length n, the number of accepting computations is no more than the number of subsets of L^{=m}. Since L is log-sparse, the number of accepting computations is $m^{O(1)} = 2^{O(n)}$. Thus L' is in FewE, and by assumption, L' is in E. Now it can easily be shown that L is in P, using techniques that must now be familiar to the reader, from Theorems 3 and 8, or from [15, 16].

We do not know if the existence of log-sparse sets in FewP-P is equivalent to the existence of sparse sets in FewP-P. We suspect that there are oracles relative to which these conditions are not equivalent.

This technique can also be used to shed light on the nature of sparse sets in UP. Define UE to be the class of sets accepted by NE machines such that for each input accepted, there is a unique accepting path. UE is just the exponential-time analogue of UP; it is easy to show that UE=E iff there are no tally sets in UP-P.

Theorem 13 UE=E iff there are no O(1)-sparse sets in UP-P.

Proof: The proof of this theorem is essentially identical to the proof of Theorem 12. However, we need the following technical notions. Define UE_k (UP_k) to be the class of languages accepted by NE (NP) machines such that if input x is accepted, then there are at most k accepting paths on input x. It follows from Theorem 3.1 of [28] that, for all k, UP=P iff $UP_k=P$. A similar proof shows that UE=E iff $UE_k=E$. For completeness, a proof of this fact is given below as Lemma 14.

The backward implication follows immediately from the fact that UE=E iff there are no tally sets in UP-P.

To prove the forward implication, assume UE=E; by Lemma 14, this implies that $UE_k = E$ for all k. Let L be an O(1)-sparse set in UP, and let k be a constant such that, for all n, there are no more than k elements of $L^{=n}$. As in the proof of Theorem 12, let L' be the set

 $\{1\langle m,j\rangle$: there are at least j strings in L of length $m\}$ \cup $\{0\langle m,j,h,l,b\rangle$: there is a sorted list of j strings in L, x_1,x_2,\ldots,x_j , each of

length m, and the lth bit of x_h is b}.

Reasoning as in the proof of Theorem 12, it is easy to see that L' is in UE_{2^k} , and thus L' is in E. Now as in the proof of Theorem 12, it is easy to see that L is in P.

The following lemma is provided merely to complete the proof of the preceding theorem. It is very similar to a result proved by Watanabe in [28].

Lemma 14 For all k, UE = E iff $UE_k = E$.

Proof: The proof is by induction on k; the basis k = 1 is trivial.

Assume $UE_{k-1} = E$, and let L be a set in UE_k , as witnessed by some NE machine N. Define L' to be the set $\{x : N \text{ has exactly } k \text{ accepting computation paths on } x\}$. It is easy to see that L' is in UE, and thus by assumption L' is in E. Thus L - L' is in UE_{k-1} , as witnessed by the machine that, on input x, rejects if $x \in L'$, and otherwise simulates N. By our inductive hypothesis, L-L' is thus in E, and thus $L = L' \cup (L - L')$ is also in E.

5 Acknowledgments

I benefited from being able to discuss this problem with a number of people, including Lane Hemachandra and Osamu Watanabe. The oracle construction is, in some ways, similar to a construction by Russell Impagliazzo and Gabor Tardos [19], and it is my

pleasure to acknowledge the discussions I had with them. The two anonymous referees provided a number of helpful comments. Finally, I would like to thank Ron Book for organizing a workshop on complexity at Santa Barbara in June, 1988, where my thoughts were turned again toward this problem.

References

- [1] E. Allender, *The complexity of sparse sets in P*, Proc. 1st Structure in Complexity Theory Conference, <u>Lecture Notes in Computer Science</u> 223, Springer-Verlag, Berlin/New York, 1986, pp. 1–11.
- [2] E. Allender, *The generalized Kolmogorov complexity of sets*, Proc. 4th Structure in Complexity Theory Conference, 1989, pp. 186–194.
- [3] E. Allender, Limitations of the upward separation technique, Proceedings of the 16th International Colloquium on Automata, Languages, and Programming, <u>Lecture Notes in Computer Science</u> 372, Springer-Verlag, Berlin/New York, 1989, pp. 18–30.
- [4] E. Allender and R. Rubinstein, P-printable sets, SIAM J. Comput 17 (1988) 1193– 1202.
- [5] E. Allender and C. Wilson, *Downward translations of equality*, to appear in Theoret. Comput. Sci.

- [6] J. Balcázar, J. Díaz, and J. Gabarró, <u>Structural Complexity I</u>, Springer-Verlag, Berlin/New York, 1988.
- [7] R. Beigel, On the relativized power of additional accepting paths, Proc. 4th Structure in Complexity Theory Conference, 1989, pp. 216–224.
- [8] A. Blass and Y. Gurevich, On the unique satisfiability problem, Information and Control 55 (1982) 80-88.
- [9] R. Book, Tally Languages and Complexity Classes, Information and Control 26 (1974) 186–193.
- [10] R. Book, C. Wilson, and Xu Mei-Rui, Relativizing time, space, and time-space, SIAM J. Comput. 11 (1982) 571–581.
- [11] J. Cai and L. Hemachandra, On the power of parity polynomial time, Proc. 6th Annual Symposium on Theoretical Aspects of Computer Science, <u>Lecture Notes in Computer Science</u> 349, Springer-Verlag, Berlin/New York, 1989, pp. 229–239.
- [12] M. Dekhtyar, On the relativization of deterministic and nondeterministic complexity classes, Proceedings of the 5th Symposium on Mathematical Foundations of Computer Science, <u>Lecture Notes in Computer Science</u> 45, Springer-Verlag, Berlin/New York, 1976, pp. 255–259.
- [13] R. Gavaldà, L. Torenvliet, O. Watanabe, and J. Balcázar, Generalized Kolmogorov complexity in relativized separations, to appear in Proc. 15th In-

- ternational Symposium on Mathematical Foundations of Computer Science, Lecture Notes in Computer Science, Springer-Verlag, Berlin/New York, 1990.
- [14] J. Hartmanis, Generalized Kolmogorov complexity and the structure of feasible computations, Proc. 24th IEEE Symposium on Foundations of Computer Science, 1983, pages 439–445.
- [15] J. Hartmanis, On sparse sets in NP-P, Information Processing Letters 16 (1983) 55–60.
- [16] J. Hartmanis, N. Immerman, and V. Sewelson. Sparse sets in NP-P: EXPTIME versus NEXPTIME, Information and Control 65 (1985) 158-181.
- [17] H. Heller, On relativized exponential and probabilistic complexity classes, Information and Control 71 (1986), 231–243.
- [18] J. Hopcroft and J. Ullman, <u>Introduction to Automata Theory</u>, <u>Languages</u>, and <u>Computation</u>, Addison-Wesley, Reading, Massachusetts, 1979.
- [19] R. Impagliazzo and G. Tardos Decision versus search in super-polynomial time, Proc. 30th Annual IEEE Symposium on Foundations of Computer Science (1989), pp. 222–227.
- [20] S. Kurtz, Sparse sets in NP-P: relativizations, SIAM J. Comput. 14 (1985) 113– 119.

- [21] J. Köbler, U. Schöning, S. Toda, and J. Torán, Turing machines with few accepting computations and low sets for PP, Proc. 4th Structure in Complexity Theory Conference, 1989, pp. 208–215.
- [22] Ming Li and Paul Vitányi, Two decades of applied Kolmogorov complexity, Proc. 3rd Structure in Complexity Theory Conference, 1988, pp. 80–101.
- [23] G. Lischke, Oracle-constructions to prove all possible relationships between relativizations of P, NP, EL, NEL, EP and NEP, Zeitschrift für mathematische Logic und Grundlagen der Mathematik 32 (1986) 257–270.
- [24] G. Lischke, Relativizations of NP and EL, strongly separating, and sparse sets, J. Information Processing and Cybernetics EIK 23 (1987) 99–112.
- [25] Wolfgang Paul, On-line simulation of k+1 tapes by k tapes requires nonlinear time, Information and Control 53 (1982) 1–8.
- [26] R. Rubinstein, Structural complexity classes of sparse sets: intractability, data compression and printability, doctoral dissertation, Northeastern University, 1988.
- [27] U. Schöning, <u>Complexity and Structure</u>, <u>Lecture Notes in Computer Science</u> 211, Springer-Verlag, Berlin/New York, 1985.
- [28] O. Watanabe, On hardness of one-way functions, Information Processing Letters 27 (1988) 151–157.
- [29] C. Wilson, Relativization, reducibilities and the exponential hierarchy, M.S. Thesis, University of Toronto, 1980.