Electronic Colloquium on Computational Complexity, Report No. 55 (2010)

Avoiding Simplicity is Complex*

Eric Allender Department of Computer Science
Rutgers University, Piscataway, NJ 08855
allender@cs.rutgers.edu

March 31, 2010

Abstract

It is a trivial observation that every decidable set has strings of length n with Kolmogorov complexity $\log n + O(1)$ if it has any strings of length n at all. Things become much more interesting when one asks whether a similar property holds when one considers *resource-bounded* Kolmogorov complexity. This is the question considered here: Can a feasible set A avoid accepting strings of low resource-bounded Kolmogorov complexity, while still accepting some (or many) strings of length n?

More specifically, this paper deals with two notions of resource-bounded Kolmogorov complexity: Kt and KNt. The measure Kt was defined by Levin more than three decades ago and has been studied extensively since then. The measure KNt is a nondeterministic analog of Kt. For all strings x, $\mathrm{Kt}(x) \geq \mathrm{KNt}(x)$; the two measures are polynomially related if and only if NEXP $\subseteq$ EXP/poly [5].

Many longstanding open questions in complexity theory boil down to the question of whether there are sets in P that avoid all strings of low Kt complexity. For example, the EXP vs ZPP question is equivalent to (one version of) the question of whether avoiding simple strings is difficult: (EXP $=$ ZPP if and only if there exist $\epsilon > 0$ and a "dense" set in P having no strings x with $\mathrm{Kt}(x) \leq |x|^{\epsilon}$ [4]).

Surprisingly, we are able to show *unconditionally* that avoiding simple strings (in the sense of KNt complexity) is difficult. Every dense set in NP $\cap$ co-NP contains infinitely many strings x such that $\mathrm{KNt}(x) \leq |x|^{\epsilon}$ for some ϵ. The proof does not relativize. As an application, we are able to show that if E $=$ NE, then accepting paths for nondeterministic exponential time machines can be found somewhat more quickly than the brute-force upper bound, if there are many accepting paths.

Key Words: Hitting Sets, Kolmogorov Complexity, Complexity Theory

1 Introduction

It has been observed before that many popular conjectures in complexity theory can be restated equivalently in terms of questions about the resource-bounded Kolmogorov

*This work has been accepted for presentation at Computability in Europe (CiE) 2010, and will appear in Springer-Verlag's Lecture Notes in Computer Science series.

ISSN 1433-8092

complexity of feasible sets, and that this restatement can serve to highlight some of the tension among these conjectures. For instance, it is common to conjecture that

1. The containment NTime$(t(n)) \subseteq$ DTime$(2^{O(t(n))})$ is nearly optimal, and that
2. Cryptographically-secure one-way functions exist.

The first of these two conjectures implies that there are polynomial time-bounded Turing machines that, for infinitely many inputs n, accept some strings in Σ^n, but none having Kt-complexity less than (say) $n/5$ [2], where Kt is a time-bounded version of Kolmogorov complexity defined by Levin [15]. (Definitions will be provided in Section 2.) In contrast, the second conjecture implies that secure pseudorandom generators exist [12], which in turn implies that any polynomial time-bounded machine *must* accept some strings in Σ^n with Kt-complexity much less than $\sqrt{n}$ *if* the machine accepts at least half of the strings in Σ^n [1]. Thus, if the popular conjectures are true, sets in P *can* avoid accepting strings with low Kt-complexity, but *only* if they don't accept many strings of any given length. If a set in P contains a lot of strings of a given input length, then it *cannot* avoid accepting some simple strings, according to the popular belief.

This paper deals with the question of how difficult it is to avoid simple strings (i.e., strings of low resource-bounded Kolmogorov complexity) while still accepting a large number of strings. The main contribution of the paper is to present one setting in which we can replace popular conjecture and vague belief with unconditional theorems, showing that easy-to-compute sets *must* contain simple strings if they contain many strings. We also present an application of this new insight to the question of whether accepting computation paths of nondeterministic exponential time machines are easy to find, assuming "only" E = NE.

Let us introduce some notation, to help us gauge how successfully a set is avoiding simple strings. For any set $A \subseteq \Sigma^*$, define $\mathrm{Kt}_A(n)$ to be $\min\{\mathrm{Kt}(x) : x \in A^{=n}\}$, where $A^{=n} = A \cap \Sigma^n$. (For a definition of Levin's measure $\mathrm{Kt}(x)$, see Section 2.) If $A^{=n} = \emptyset$, then $\mathrm{Kt}_A(n)$ is undefined. Note that the rate of growth of $\mathrm{Kt}_A(n)$ is a measure of how successfully A avoids strings of low Kt complexity. The rate of growth of $\mathrm{Kt}_A(n)$ for sets A in P and P/poly is especially of interest, as can be seen from the following theorem. (We give a more precise definition of "dense" in Section 3, but for now it is sufficient to consider a set to be "dense" if it contains at least $2^n/n$ strings of each length n. An "NE search problem" is the task of mapping an input x to an accepting computation of M on input x, if one exists, where M is an NE machine.)

Theorem 1

- *There is an* NE *search problem that is not solvable in time* $2^{2^{o(n)}}$ *if and only if there is a set* $A \in$ P *and an* $\epsilon > 0$ *such that* $\mathrm{Kt}_A(n) \neq O(n^\epsilon)$ *[2, Theorem 6].*
- *There is an* NE *search problem that is not solvable in time* $2^{O(n)}$ *if and only if there is a set* $A \in$ P *such that* $\mathrm{Kt}_A(n) \neq O(\log n)$ *[2, Theorem 6].*
- EXP $\not\subseteq$ P/poly *if and only if for every dense set* $A \in$ P/poly *and every* $\epsilon > 0$, $\mathrm{Kt}_A(n) = O(n^\epsilon)$ *[3, Theorem 12].*

- *There is a set $B \in$ E and an $\epsilon > 0$ such that, for all large n, there is no circuit of size $2^{\epsilon n}$ accepting $B^{=n}$ if and only if for every dense set $A \in$ P/poly, $\mathrm{Kt}_A(n) = O(\log n)$ [3, Theorem 13].*

- EXP $\neq$ ZPP *if and only if for every dense set $A \in$ P and every $\epsilon > 0$, $\mathrm{Kt}_A(n) = O(n^\epsilon)$ [4].*

A nondeterministic analog of Levin's Kt measure, denoted KNt, was introduced recently [5]. For any set A, let $\mathrm{KNt}_A(n)$ be $\min\{\mathrm{KNt}(x) : x \in A^{=n}\}$. The rate of growth of $\mathrm{KNt}_A(n)$ is similarly related to open questions in complexity theory:

Theorem 2 *[5, Theorem 44] There is a set $B \in \mathrm{NE/lin}$ such that for all large n, there is no nondeterministic circuit of size $2^{\epsilon n}$ accepting $B^{=n}$ if and only if for every dense set A in* NP/poly $\cap$ coNP/poly, $\mathrm{KNt}_A(n) = O(\log n)$.

Theorem 2 presents a condition regarding $\mathrm{KNt}_A(n)$ for dense sets A in a *nonuniform* class, and Theorem 1 contains analogous conditions regarding dense sets in both uniform *and* nonuniform classes. It is natural to wonder if Theorem 2 can be extended, to say something about the corresponding uniform class, and the experience of Theorems 1 and 2 could lead one to expect that such an extension would consist of showing that a statement about the KNt-complexity of dense sets in NP $\cap$ co-NP is equivalent to some longstanding open question in complexity theory.

Thus it is of interest that our main theorem shows *unconditionally* that $\mathrm{KNt}_A(n)$ grows slowly for all dense sets in NP $\cap$ co-NP (and even for all of those dense sets lying in $(\mathrm{NP} \cap \text{co-NP})/n^{o(1)}$).

The rest of the paper is organized as follows. Definitions and preliminaries are presented in Section 2. The main results are presented in Section 3. An application to NE search problems is presented in Section 4. And finally, some musings about possible improvements to the main result are presented in Section 5.

2 Preliminaries

We assume that the reader is familiar with complexity classes such as P, ZPP, NP, AM, and PSPACE; for background consult a standard text such as [6]. We use the following notation for deterministic and nondeterministic exponential-time complexity classes: E = DTime$(2^{O(n)})$, NE = NTime$(2^{O(n)})$, EXP = DTime$(2^{n^{O(1)}})$, and NEXP = NTime$(2^{n^{O(1)}})$. $\mathrm{P}^{\mathrm{NP}[n]}$ is the class of languages accepted by polynomial-time oracle Turing machines with an oracle from NP, where the oracle machine makes at most n oracle queries on inputs of length n.

For any complexity class $\mathcal{C}$, and function $h(n) : \mathbb{N} \to \mathbb{N}$, let $\mathcal{C}/h(n)$ denote the class of sets B such that, for some "advice function" $a(n) : \mathbb{N} \to \Sigma^{h(n)}$, and some set $A \in \mathcal{C}$, $x \in B$ if and only if $(x, a(|x|)) \in A$. $\mathcal{C}$/poly denotes $\bigcup_k \mathcal{C}/n^k + k$; $\mathcal{C}$/lin denotes $\bigcup_k \mathcal{C}/kn$. The class P/poly has an equivalent characterization as the class of problems solvable by families of polynomial-size circuits. Note in particular that $(\mathrm{NP} \cap \text{co-NP})$/poly is quite possibly a proper subset of NP/poly $\cap$ coNP/poly.

Levin defined $\mathrm{Kt}(x)$ to be $\min\{|d| + \log t : U(d) = x \text{ in time } t\}$ [15], where U is some fixed universal Turing machine. (It is important to note that Levin's definition is *independent* of any run-time t; the "t" that appears in the definition is a quantity that participates in the minimization expression.) Later, it was observed that $\mathrm{Kt}(x)$ is polynomially related to the oracle circuit size that is required to compute the function that has x as its truth table [5], where the oracle is a complete set for E. In order to obtain a time-bounded notion of Kolmogorov complexity in the spirit of Levin's Kt function that is related to circuit complexity for more general oracles (including the empty oracle), a new measure, called KT, was defined [4]:

Definition 1 *Let U be a universal Turing machine and let B be an oracle. Define the measure* $\mathrm{KT}^B(x)$ *to be*

$$\mathrm{KT}^B(x) = \min\{\, |d| + t \;:\; U^{B,d} \textit{ describes } x \textit{ in time } t,\ (\textit{meaning that } \forall b \in \{0,1,*\}\ \forall i \le |x|+1, U^{B,d}(i,b) \textit{ accepts in } t \textit{ steps iff } x_i = b)\}.$$

(The notation "$U^{B,d}(i,b)$" indicates that the machine U has random access (or "oracle access") to both the string d and the oracle B. This allows the running time to be less than $|d|$.) We omit the superscript B if $B = \emptyset$.

It is known that one can pick a complete set C for E such that Levin's definition of $\mathrm{Kt}(x)$ is linearly-related to $\mathrm{KT}^C(x)$ [4].

A nondeterministic analog of Kt called KNt was recently investigated [5], and it was shown that $\mathrm{KNt}(x)$ is linearly related to $\mathrm{KT}^D(x)$ for a set D that is complete for NE. Thus, for this paper, we will let $\mathrm{Kt}(x)$ and $\mathrm{KNt}(x)$ denote $\mathrm{KT}^C(x)$ and $\mathrm{KT}^D(x)$ for this E-complete set C and NE-complete set D, respectively.

For a given set A, and oracle B, we define $\mathrm{KT}^B_A(n)$ to be equal to $\min\{\mathrm{KT}^B_A(x) : x \in A^{=n}\}$. Thus $\mathrm{Kt}_A(n) = \mathrm{KT}^C_A(n)$, and $\mathrm{KNt}_A(n) = \mathrm{KT}^D_A(n)$.

We assume that the reader is familiar with polynomial-time Turing reducibility, denoted $\leq^{\mathrm{p}}_T$. We also need to make use of reductions computed by polynomial-size *circuits*, instead of polynomial-time *machines*. A P/poly-Turing reduction of a set A to a set B is a family of polynomial-size circuits computing A, where the circuits have *oracle gates* for B, in addition to the usual AND and OR gates. (An oracle gate for B outputs 1 if the string y that is presented to it as input is an element of B.) If a P/poly-Turing reduction has the property that there is no path in the circuit from one oracle gate to another, then it is called a P/poly-truth-table reduction, denoted $\leq^{\mathrm{P/poly}}_{tt}$.

3 Main Result

The main theorem applies only to languages that have "sufficiently many" strings; we call such sets "dense". The following definition makes precise exactly what sort of "density" is required:

Definition 2 *A set $A \subseteq \{0,1\}^*$ is* dense *if there is a k such that for every n there is some $m, n \le m \le n^k + k$ such that $|A \cap \{0,1\}^m| \ge 2^m/m^k$.*

Theorem 3 *Let A be a dense set in* $(\text{NP} \cap \text{co-NP})/a(n)$ *for some* $a(n) = n^{o(1)}$. *Then for every* $\epsilon > 0$ *there are infinitely many* $x \in A$ *such that* $\text{KNt}(x) < |x|^{\epsilon}$.

Proof: Most of the work has already been done in an earlier paper, in which it was shown that R_{KNt} (the set of "KNt-random strings", i.e., the set of strings x such that $\text{KNt}(x) \geq |x|$) is not in NP $\cap$ co-NP [5]. It was noticed only shortly after that paper was submitted for publication that the lower bound applied not only to R_{KNt}, but in fact to *every* dense set A such that, for some $\epsilon > 0$, $\text{KNt}_A(n) \neq \Omega(n^{\epsilon})$. We need to recall some of the main theorems of earlier work on this topic.

One of the main insights obtained in earlier work is that for "large" complexity classes, dense sets having only strings of high Kolmogorov complexity are hard under P/poly reductions. The following definition captures the property that a "large" class needs to have, in order for the proof to go through:

Definition 3 *A set B is* PSPACE*-robust if* $\text{P}^B = \text{PSPACE}^B$.

The notion of PSPACE-robustness was defined by Babai et al [8], who observed that every set that is complete for EXP under $\leq_T^{\text{P}}$ reductions is PSPACE-robust. Later, it was shown that NEXP also has this property [5].

Theorem 4 *[4, Theorem 31] Let B be any* PSPACE*-robust set. Let A be a set such that for some $\epsilon > 0$ and k, for every n there is some m such that*

- $n \leq m \leq n^k + k$,
- $|A \cap \{0,1\}^m| \geq 2^m/m^k$
- $\text{KT}_A^B(m) \geq m^{\epsilon}$.

Then B is reducible to A via $\leq_{tt}^{\text{P/poly}}$ reductions.

(This is a slight modification of the statement of the theorem as given in [4]. There, it was assumed that A contains many strings of *every* length, and contains *no* strings of low KT^B complexity. However, the $\leq_{tt}^{\text{P/poly}}$ reduction that is given in [4] has the property that, on inputs of length n, all queries to the oracle A have the same length m, and the reduction works properly as long as, for the given length m, A contains many strings and no strings of low KT^B complexity. Thus, by simply encoding the length m into the nonuniform advice of the $\leq_{tt}^{\text{P/poly}}$ reduction, the proof given in [4] suffices to establish Theorem 4.) We remark also that the proof given in [4] proceeds by showing that A can be used as a test to distinguish truly random strings from pseudorandom strings produced by a pseudorandom generator constructed from B. Thus, the same argument shows that B is reducible to A even if A contains only a *few* strings with low KT^B complexity. Consequently, it is possible to improve the statement of Theorem 3 to say that every dense set in $(\text{NP} \cap \text{co-NP})/a(n)$ has *many* strings of KNt complexity $\leq n^{\epsilon}$, for infinitely many n. We do not pursue that generalization here.

Since we are using the definition of KNt as KT^D for some set D that is complete for NE, and since every set that is complete for NE is PSPACE-robust, Theorem 4 immediately yields the following corollary:

Corollary 5 *Let A be any dense set such that* $\mathrm{KNt}_A(n) = \Omega(n^\epsilon)$ *for some* $\epsilon > 0$. *Then* A *is hard for* NEXP *under* $\leq_{tt}^{\mathrm{P/poly}}$ *reductions.*

We also need to use the fact that any A that satisfies the hypothesis of Corollary 5 is also hard for PSPACE under probabilistic reductions:

Theorem 6 *[4, Theorem 33 and Lemma 35] Let A be any set of polynomial density, such that* $\mathrm{Kt}_A(n) = \Omega(n^\epsilon)$ *for some* $\epsilon > 0$. *Then* PSPACE $\subseteq$ ZPP^A.

Note that the term "polynomial density" as used in [4] is slightly more restrictive than the term "dense" as defined in this paper, since a set has "polynomial density" if it contains many strings of *every* length.

Corollary 7 *Let A be any dense set in* $(\mathrm{NP} \cap \text{co-NP})/a(n)$ *such that* $\mathrm{KNt}_A(n) = \Omega(n^\epsilon)$ *for some* $\epsilon > 0$. *Then* PSPACE $\subseteq \bigcup_k (\mathrm{NP} \cap \text{co-NP})/2a(n^k) + O(\log n)$.

Proof: Note that $\mathrm{Kt}_A(n) \geq \mathrm{KNt}_A(n) = \Omega(n^\epsilon)$. Thus we would like to modify the proof of Theorem 6 to (nearly) obtain that PSPACE $\subseteq$ ZPP^A.

The proof of Theorem 6 given in [4] presents a ZPP reduction with the property that, on inputs of length n, there are lengths m_1 and m_2 such that all queries to the oracle have length either m_1 or m_2. (Queries to length m_1 are used to obtain a string of high complexity, which is then used in conjunction with the Impagliazzo-Wigderson construction [14] to derandomize a BPP^A reduction, which only makes queries of length m_2.) The length m_2 can be replaced by any m_2' such that $m_2 \leq m_2' \leq m_2^{O(1)}$, as long as the reduction is given suitable advice, saying which length m_2' has sufficiently many strings, and the length m_1 can be replaced by any m_1' at most polynomially larger than m_2', again if the reduction is given advice, saying which length m_1' is suitable. Thus the ZPP reduction running in time n^k can be simulated by a $(\text{ZPP}^{\mathrm{NP} \cap \text{co-NP}})/2a(n^c) + O(\log n)$ computation, where c depends on k and on the density parameters of A. The corollary follows by observing that ZPP$^{\mathrm{NP} \cap \text{co-NP}} =$ NP $\cap$ co-NP. □□

We now proceed with the proof of Theorem 3. The proof is by contradiction: Assume that A is a dense set in $(\mathrm{NP} \cap \text{co-NP})/a(n)$ for some $a(n) = n^{o(1)}$ such that, for all large $x \in A$, we have $\mathrm{KNt}(x) \geq |x|^\epsilon$. That is, $\mathrm{KNt}_A(n) = \Omega(n^\epsilon)$.

By Corollaries 5 and 7 we have PSPACE $\subseteq (\mathrm{NP} \cap \text{co-NP})/n^{o(1)}$ and NEXP $\subseteq$ P^A/poly $\subseteq$ P$^{(\mathrm{NP} \cap \text{co-NP})/a(n)}$/poly = $(\mathrm{NP} \cap \text{co-NP})/\mathrm{poly}$.

It is known that if NEXP $\subseteq (\mathrm{NP} \cap \text{co-NP})/\mathrm{poly}$ then NEXP = AM [5, Theorem 29]. Thus under our assumptions we have

$$\mathrm{NEXP} = \mathrm{AM} = \mathrm{PSPACE} \subseteq (\mathrm{NP} \cap \text{co-NP})/n^{o(1)}.$$

This is a contradiction, since $(\mathrm{NP} \cap \text{co-NP})/n^{o(1)} \subseteq \mathrm{P}^{\mathrm{NP}[n]}/n$, and it was shown by Buhrman, Fortnow, and Santhanam [10] that NEXP is not contained in $\mathrm{P}^{\mathrm{NP}[n]}/n$. □□

It would be nice to know if a better upper bound on the KNt-complexity of dense sets in NP $\cap$ co-NP (or in P) can be proved.

3.1 Does This Relativize?

The proof of Theorem 3 does not relativize, since it relies on Theorem 6 (which in turn relies on the characterization of PSPACE in terms of interactive proofs [16, 17]) and also Theorem 28 of [5] (which relies on the characterization of NEXP in terms of interactive proofs [7]). However, we do not know if the statement of Theorem 3 actually fails relative to some oracle.

Spakowski [18] has pointed out that an oracle construction of Buhrman, Fortnow, and Laplante [9] might be relevant. They present a set A such that $\text{CND}^{2^{\sqrt{|x|}}}(x) \geq |x|/4$ for all $x \in A$, where $\text{CND}^{2^{\sqrt{n}}}$ is a notion of "$2^{\sqrt{n}}$-time-bounded nondeterministic distinguishing complexity". It is known that CND-complexity is related to KNt complexity [5], and one can easily show that, for their set $A \in \text{P}^A$, there is some $\epsilon > 0$ such that $\text{KNt}^A(x) \geq |x|^\epsilon$ for all $x \in A$, where $\text{KNt}^A(x)$ is the measure that results when one defines KNt complexity using a universal Turing machine that can access the oracle A. Spakowski suggests that a slight modification of their construction yields a set A satisfying the same conditions, that contains many strings of length n, for infinitely many n. Thus this comes close to being an oracle relative to which Theorem 3 fails.

4 An Application to Search Problems

One of the aspects of the theory of NP-completeness that makes it so widely applicable, is the fact that, for NP-complete problems, *search* is equivalent to *decision*. That is, the problem of *deciding* membership in an NP-complete set is polynomially-equivalent to the problem of *finding* a proof of membership. Hartmanis, Immerman, and Sewelson observed that the proof of equivalence that works for NP-complete problems breaks down for exponential-time computations, and they asked whether search and decision are also equivalent for NE-complete problems [11]. A partial answer was provided by Impagliazzo and Tardos [13], who presented an oracle relative to which E = NE but relative to which there exists a nondeterministic exponential-time machine M such that there is no function computable in exponential time that maps each input x accepted by M to an accepting computation of M on input x. An alternative oracle construction was subsequently given by Buhrman, Fortnow, and Laplante [9].

The trivial brute-force deterministic algorithm for finding accepting computations of NE machines takes doubly exponential time $2^{2^{O(n)}}$. No significantly better upper bound is known, even for the special case of finding accepting computations of probabilistic NE machines, that have *many* accepting computation paths if they have any at all. This has been the case, even under the assumption that E = NE.

As a consequence of the results of Section 3, we can now say something nontrivial about an upper bound on the complexity of finding accepting computations of NE machines if E = NE – at least for certain classes of NE machines. (Actually, it suffices to use the weaker assumption that NEXP $\subseteq$ EXP/poly.) Let ZPE be the exponential-time analog of the complexity class ZPP. That is, B is in ZPE if there are two nondeterministic Turing machines M_0 and M_1 running for time 2^{cn} for some c, where M_0 accepts $\overline{B}$ and M_1 accepts B, with the property that if $x \in B$, then for at least half of the strings

r of length 2^{cn}, M_1 accepts x along the computation path given by r, and if $x \notin B$, then for at least half of the strings r M_0 accepts x along the computation path given by r. Thus, for every string x, either half of the strings r of length $2^{c|x|}$ are accepting computations of M_1, or half of the strings r are accepting computations of M_0. A ZPE *search problem* (defined by the machines M_0 and M_1) is the task of taking x as input, and producing a string r as output, that causes either M_0 or M_1 to accept.

Theorem 8 *If* NEXP $\subseteq$ EXP/poly, *then for every* ZPE *search problem, there is a deterministic algorithm M solving it with the property that, for every $\epsilon > 0$, M runs in time $2^{2^{|x|^\epsilon}}$ for infinitely many x.*

Proof: Consider a ZPE search problem defined by machines M_0 and M_1. Let N be an NE machine running in time 2^{cn} that, on input x, guesses a string r of length 2^{cn} and accepts if r causes either M_0 or M_1 to accept on input x. (Note that N accepts *every* string x.)

Let $d : \mathbb{N} \rightarrow \{0,1\}^*$ be a standard bijection (e.g., $d(i)$ is the string x such that the binary representation of $i+1$ is $1x$). Let W_N be the set $\{r : |r| = n^{c+1}$ and (some prefix of) r causes N to accept the string $d(n)\}$. Note that, since $|d(n)| = O(\log n)$, W_N is in P, and A is dense (since it contains at least half of the strings of each length of the form n^{c+1}).

By Theorem 3, for every $\epsilon > 0$ there are infinitely many $r \in W_N$ such that $\mathrm{KNt}(x) < |r|^\epsilon$. Since we are assuming that NEXP $\subseteq$ EXP/poly, it follows that Kt and KNt are polynomially related [5], and thus we have that for every $\epsilon > 0$ there are infinitely many $r \in W_N$ such that $\mathrm{Kt}(r) < |r|^\epsilon$. Let C be the E-complete set such that $\mathrm{Kt}(x) = \mathrm{KT}^C(x)$.

Consider the following algorithm M: On input x, compute n so that $d(n) = x$. For $k = 1$ to n^c, for all descriptions d of length k, see if $U^{C,d}$ describes a string r of length n^c in time k. If so, and if r causes N to accept on input x, then halt and output r.

It is straightforward to verify that the algorithm M has the properties claimed for it in the statement of the theorem. □□

The conclusion of Theorem 8 holds for a great many more NE search problems than merely those in ZPE. It holds for any NE machine N for which the language W_N constructed in the proof of Theorem 8 is dense. (This corresponds to those problems in NE that are accepted by NE machines that have many accepting computation paths for at least one string of every length (or, more generally, at least one string out of every $O(1)$ consecutive lengths).) Rather than creating a new definition to capture this class, we simply state the following corollary:

Corollary 9 *If* NEXP $\subseteq$ EXP/poly, *then for every* NE *search problem defined by an* NE *machine N such that the set W_N is dense, there is a deterministic algorithm M solving it with the property that, for every $\epsilon > 0$, M runs in time $2^{2^{|x|^\epsilon}}$ for infinitely many x.*

It is natural to wonder if E = NE implies faster algorithms for *all* instances of ZPE, instead of merely for infinitely many inputs x. This is essentially a question of whether polynomial-time computations can accept many strings while avoiding all simple strings for *some* input lengths, but not for others. This topic is discussed at greater length in the next section.

5 Are Σ^n and Σ^m Fundamentally Different, for $n \neq m$?

In this section, we discuss the KNt complexity of dense sets in P. Theorem 3 says that, for every dense set $A \in$ P, there exist infinitely many lengths n such that A contains a string of length n having KNt complexity less than n^ϵ. In this section, we observe that we can essentially swap the quantifiers. There are long segments of consecutive input lengths $S = [i, i+1, \ldots, i^c]$ such that, for every $m \in S$, *every* machine running in time m^k must accept strings in Σ^m with KNt complexity at most m^ϵ if it accepts many strings of length m. There may be long "deserts" of input lengths where, for all we know, polynomial-time machines can behave badly by avoiding all of the simple strings while still accepting many strings. However, we are guaranteed that there are infinitely many large "oases" in the desert, where machines behave as expected (i.e., by accepting some strings with low KNt complexity, if they accept many strings).

Consider the standard universal set for DTime(n^k): $A_k = \{(i, x) : M_i \text{ accepts } x \text{ in } |x|^k \text{ steps}\}$, where we assume an enumeration of machines such that M_0 accepts Σ^* in linear time. Let $a(n)$ be defined to be the index $i \leq n$ of the machine M_i such that, among all of the machines M_j with $j \leq n$ that run in time n^k on inputs of length n and accept at least $2^n/n^k$ strings of length n, $\text{KNt}_{L(M_j)}(n)$ is maximized. Note that $a(n)$ is always defined, by our choice of M_0. The set $S_k = \{x : M_{a(|x|)}(x) = 1\}$ is in P/$\log n$ and contains at least $2^n/n^k$ strings of each length n, and has KNt complexity asymptotically as high as any dense set in DTime(n^k).

Define the *k-oasis* to be the set $\{n : \text{KNt}_{S_k}(n) \leq n^{1/k}\}$. It is immediate that the $(k+1)$-oasis is a subset of the k-oasis, for every k. Also note that, for every $c \geq 1$, and every k, the k-oasis contains infinitely many sequences of consecutive numbers $n, n+1, \ldots, n^c$, since otherwise the set S_k would be a dense set in P/$\log n$ that would be hard for NEXP under $\leq_{tt}^{\text{P/poly}}$ reductions (by Theorem 4) and would also be hard for PSPACE under ZPP/$O(\log n)$ reductions (by Corollary 5), and one would obtain a contradiction exactly as in the proof of Theorem 3.

That is, the k-oases for large k contain superpolynomially-long sequences of consecutive input lengths where all DTime(n^k) machines "behave well", and the k-oases for smaller values of k are even larger.

Since there is not a recursive enumeration of pairs of machines that define sets in NP $\cap$ co-NP, the strategy that was used in defining "k-oases" for DTime(n^k) must be modified, in order to define an analogous notion of k-oasis for the classes NTime(n^k)$\cap$ coNTime(n^k). It suffices to make use of a *nonrecursive* enumeration of pairs of machines; details will appear in the full version of this paper.

It seems reasonable to conjecture that each k-oasis is actually $\mathbb{N}$ (or at least, that it contains all large natural numbers). Otherwise, for infinitely many lengths m, there are circuits of size m^k that accept a large fraction of the strings of length m, but accept nothing of small KNt complexity, while this is impossible for other lengths m'. This would seem to indicate that Σ^m has some structural property that small circuits are able to exploit, whereas $\Sigma^{m'}$ has no such structure. However, Σ^m seems *devoid* of any useful structure that is not shared by $\Sigma^{m'}$ for $m' \neq m$.

6 Closing Comments

For sufficiently "powerful" forms of resource-bounded Kolmogorov complexity (such as KT^E where E is complete for EXPSPACE), the lexicographically first element of $A^{=n}$ will always have logarithmic complexity, for any $A \in \mathrm{P}$ [5]. Conceivably, one could define a version of resource-bounded Kolmogorov complexity related to a low level of the exponential-time hierarchy (with just a few alternations – and therefore conceptually "closer" to KNt than KT^E) where this same technique could be applied. It seems unlikely that KNt is powerful enough to always give the lexicographically least element of $A^{=n}$ logarithmic complexity, for every set A in P, although we know of no unlikely consequences, if that were to be the case.

Acknowledgments

The research of the author is supported in part by NSF Grants DMS-0652582, CCF-0830133, and CCF-0832787. Some of this work was performed while the author was a visiting scholar at the University of Cape Town and at the University of South Africa. We acknowledge helpful discussions with Chris Umans, Holger Spakowski, and Ronen Shaltiel.

References

[1] E. Allender. Some consequences of the existence of pseudorandom generators. *Journal of Computer and System Sciences*, 39:101–124, 1989.

[2] E. Allender. Applications of time-bounded Kolmogorov complexity in complexity theory. In *O. Watanabe (Ed.), Kolmogorov Complexity and Computational Complexity*, pages 4–22. Springer, 1992.

[3] E. Allender. When worlds collide: Derandomization, lower bounds, and Kolmogorov complexity. In *Proc. Conf. on Found. of Software Technology and Theo. Comp. Sci. (FST&TCS)*, volume 2245 of *Lecture Notes in Computer Science*, pages 1–15, 2001.

[4] E. Allender, H. Buhrman, M. Koucký, D. van Melkebeek, and D. Ronneburger. Power from random strings. *SIAM Journal on Computing*, 35:1467–1493, 2006.

[5] E. Allender, M. Koucký, D. Ronneburger, and S. Roy. The pervasive reach of resource-bounded Kolmogorov complexity in computational complexity theory. *Journal of Computer and System Sciences*. To appear.

[6] Sanjeev Arora and Boaz Barak. *Computational Complexity, a modern approach*. Cambridge University Press, 2009.

[7] L. Babai, L. Fortnow, and C. Lund. Non-deterministic exponential time has two-prover interactive protocols. *Computational Complexity*, 1:3–40, 1991.

[8] L. Babai, L. Fortnow, N. Nisan, and A. Wigderson. BPP has subexponential time simulations unless EXPTIME has publishable proofs. *Computational Complexity*, 3:307–318, 1993.

[9] H. Buhrman, L. Fortnow, and S. Laplante. Resource-bounded Kolmogorov complexity revisited. *SIAM Journal on Computing*, 31(3):887–905, 2002.

[10] Harry Buhrman, Lance Fortnow, and Rahul Santhanam. Unconditional lower bounds against advice. In *Proc. of International Conference on Automata, Languages, and Programming (ICALP)*, volume 5555 of *Lecture Notes in Computer Science*, pages 195–209, 2009.

[11] Juris Hartmanis, Neil Immerman, and Vivian Sewelson. Sparse sets in NP-P: EXPTIME versus NEXPTIME. *Information and Control*, 65(2/3):158–181, 1985.

[12] J. Håstad, R. Impagliazzo, L. Levin, and M. Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28:1364–1396, 1999.

[13] R. Impagliazzo and G. Tardos. Decision versus search problems in superpolynomial time. In *Proc. IEEE Symp. on Found. of Comp. Sci. (FOCS)*, pages 222–227, 1989.

[14] R. Impagliazzo and A. Wigderson. $P = BPP$ if E requires exponential circuits: Derandomizing the XOR lemma. In *Proc. ACM Symp. on Theory of Computing (STOC) '97*, pages 220–229, 1997.

[15] L. A. Levin. Randomness conservation inequalities; information and independence in mathematical theories. *Information and Control*, 61:15–37, 1984.

[16] C. Lund, L. Fortnow, H. Karloff, and N. Nisan. Algebraic methods for interactive proof systems. *Journal of the ACM*, 39:859–868, 1992.

[17] A. Shamir. IP = PSPACE. *Journal of the ACM*, 39:869–877, 1992.

[18] Holger Spakowski. Personal communication. 2009.

ECCC ISSN 1433-8092

http://eccc.hpi-web.de