

# When Worlds Collide: Derandomization, Lower Bounds, and Kolmogorov Complexity

Eric Allender

Department of Computer Science  
Rutgers University  
Piscataway, NJ 08854-8019, USA  
[allender@cs.rutgers.edu](mailto:allender@cs.rutgers.edu)  
<http://www.cs.rutgers.edu/~allender>

©Springer-Verlag

**Abstract.** This paper has the following goals:

- To survey some of the recent developments in the field of derandomization.
- To introduce a new notion of time-bounded Kolmogorov complexity (KT), and show that it provides a useful tool for understanding advances in derandomization, and for putting various results in context.
- To illustrate the usefulness of KT, by answering a question that has been posed in the literature, and
- To pose some promising directions for future research.

## 1 Introduction

If one were to have taken a poll among complexity theoreticians in 1985, asking if probabilistic algorithms are more powerful than deterministic ones (i.e., if  $P = BPP$ ) I feel sure that a clear majority would have thought it more likely that  $BPP \neq P$ . Similarly, during the late 1980's when the “Arthur-Merlin” classes MA and AM were introduced [BM88] it is fair to say that most people in the community thought it most likely that MA and AM were *not* merely new names for NP. A poll taken today would, I believe, show a majority believing  $BPP = P$  and  $MA = AM = NP$ .

Such swings of opinion are a sure sign of important progress in the field. (Consider the likely outcome of taking similar polls about whether  $NL = coNL$ , or whether  $IP = PSPACE$ , before and after the announcements that these problems had been settled [Imm88, Sze88, LFKN92, Sha92].) In the case of BPP, MA, and AM, our new understanding of these classes can be credited to advances in the field of *derandomization* – a field that is so large and active that I will not pretend to be able to survey it here.

Rather, I will focus on a few recent and exciting developments in the field of derandomization, and I will show how these developments can be understood

in the context of *time-bounded Kolmogorov complexity*. This has the additional benefit of shedding light on the process of trying to prove lower bounds on the circuit complexity of various computational problems. In particular, I will highlight connections that exist between derandomization, Kolmogorov complexity, and the *natural proofs* framework of [RR97].

In this article, most proofs are sketched or omitted. A more detailed article describing this work and related results is in preparation with Detlef Ronneburger.

## 2 Ancient History

I assume that the reader is familiar with the complexity classes P, NP, MA, AM, and BPP. As has become customary, E and NE refer to  $\text{DTIME}(2^{O(n)})$  and  $\text{NTIME}(2^{O(n)})$ , respectively, whereas EXP and NEXP refer to  $\text{DTIME}(2^{n^{O(1)}})$  and  $\text{NTIME}(2^{n^{O(1)}})$ , respectively. Computations performed by circuits of AND, OR, and NOT gates having depth  $O(1)$  and polynomial size define the class  $\text{AC}^0$ . When I refer to  $\text{AC}^0$  I shall always refer to DLOGTIME-Uniform  $\text{AC}^0$  (sometimes also called  $U_E$ -uniform  $\text{AC}^0$ ) unless I explicitly mention non-uniform circuits. Occasionally, it is useful to refer to circuit size bounds other than polynomial. For instance,  $\text{AC}^0(2^{\epsilon n})$  refers to the class of languages accepted by circuit families of AND, OR, and NOT gates having depth  $O(1)$  and size bounded by  $2^{\epsilon n}$ . P/poly is the class of languages for which there exist circuit families of polynomial size (with no restriction on circuit depth). For background on complexity classes and uniformity, consult standard texts such as [Pap94,DK00,Vol99].

A set  $L$  is *P-printable* if there is a function that, on input  $n$ , produces in time  $n^{O(1)}$  a list of all elements of  $L$  having length at most  $n$ . P-printable sets were defined in [HY84] and were studied in [AR88] and elsewhere.

It remains an open question if NEXP is contained in P/poly. Perhaps more surprisingly, it remains unknown if there exists some  $\epsilon < 1$  such that NEXP is contained in  $\text{AC}^0(2^{\epsilon n})$ . (The lower bounds for the PARITY and MAJORITY functions [Hås88] provide examples of problems in P that lie outside of non-uniform  $\text{AC}^0(2^{n^\epsilon})$ , but in order to find examples of problems requiring larger circuits, currently we are forced to look outside of NEXP.)

The exponential-time analog of the P=NP problem is the question of whether or not EXP=NEXP. The reader probably knows that P=NP if and only if accepting computation paths of NP machines can be found in polynomial time. The situation at the level of exponential time is less clear, however. This motivates the following definition.

A *NEXP search problem* is a problem where, for a given nondeterministic Turing machine running in time  $2^{n^c}$  for some  $c$ , one takes as input a string  $x$  and produces as output an accepting computation of  $M$  on input  $x$  (if such a string exists). (In the case where the running time is  $2^{O(n)}$  we call this an *NE search problem*.) It is clear that if every NEXP search problem is solvable in deterministic exponential time, then NEXP = EXP; however the converse is not known to hold [AW90,IT89,BFL01].

## 2.1 Levin's Kt Complexity

There can be no doubt that the question of how to find accepting computations of a nondeterministic machine is of central importance to computer science. Levin observed that there is an easy-to-compute ordering on  $\Sigma^*$  giving an essentially-optimal search strategy [Lev84]. This can be stated more formally by means of Levin's formulation of *time-bounded Kolmogorov complexity*.

**Definition 1 (Levin).** Let  $U$  be a universal Turing machine. Define  $\text{Kt}(x)$  to be  $\min\{|d| + \log t : U(d) = x \text{ in at most } t \text{ steps}\}$ .

Note that  $\log |x| \leq \text{Kt}(x) \leq |x| + O(\log |x|)$ .

The elements of  $\Sigma^*$  can be enumerated in order of increasing  $\text{Kt}(x)$ , and Levin observed that this ordering yields essentially the fastest way to search for accepting computations. The following definitions (from [All89,All92]) will be useful in stating certain correspondences.

**Definition 2.** Let  $L \subseteq \Sigma^*$ . Define  $\text{Kt}_L(n)$  to be  $\min\{\text{Kt}(x) : x \in L^n\}$ .

(In [All89,All92] the function  $\text{Kt}_L$  was called  $K_L$ .) Here and elsewhere,  $L^n$  is the set of all strings in  $L$  of length  $n$ . If there is no string in  $L$  of length  $n$ , then  $\text{Kt}_L(n)$  is undefined. When considering the rate of growth of a function  $\text{Kt}_L(n)$ , the undefined values are not taken into consideration. Observe that, for every language  $L$ ,  $\log n \leq \text{Kt}_L(n) \leq n + O(\log n)$ .

At this point, we can state some connections between the complexity of NEXP search problems, and time-bounded Kolmogorov complexity. The following statements are almost identical to observations in [All89,All92], and they derive from Levin's original insight about Kt-complexity.

**Theorem 1.** *The following are equivalent:*

- All NEXP search problems are EXP-solvable.
- For all  $L$  in P,  $\text{Kt}_L(n) = \log^{O(1)} n$ .

**Theorem 2.** *The following are equivalent:*

- For every  $\epsilon > 0$ , all NE search problems are solvable in time  $2^{2^{\epsilon n}}$ .
- For every  $\epsilon > 0$  and for every  $L$  in P,  $\text{Kt}_L(n) = O(n^\epsilon)$ .

Since most complexity theoreticians conjecture that NEXP requires doubly exponential time, it follows that most complexity theoreticians conjecture that there are languages  $L$  in P such that  $\text{Kt}_L(n)$  grows nearly linearly.

On the other hand, most complexity theoreticians also conjecture that pseudorandom generators exist. As the following paragraphs explain, it follows that most complexity theoreticians need to conjecture that  $\text{Kt}_L(n)$  grows *slowly* if  $L$  is in P (or even in P/poly) and has very many strings in it.

For this paper, it will not be necessary to give precise definitions of pseudorandom generators. (The cited references provide details for the interested reader.) Recall only that a pseudorandom generator takes a short *seed* as input,

and produces a long *pseudorandom* output. In many applications we are interested in the case where pseudorandom output of length  $n$  is produced in time polynomial in  $n$ . In this case, note that the output of a pseudorandom generator is a string of small Kt-complexity.

A pseudorandom generator is said to be *secure* if, for any circuit  $C$  of size  $s$ , the probability that  $C$  accepts a random string of length  $n$  is within  $\epsilon$  of the probability that  $C$  accepts a pseudorandom string of length  $n$ . (The parameters  $s$  and  $\epsilon$  vary according to the security requirements.) Thus in particular note that if one has a secure pseudorandom generator and a circuit of polynomial size that accepts at least  $2^n/n$  inputs of length  $n$  (i.e., the circuit accepts a “dense” subset of  $\Sigma^n$ ), then it should also accept quite a few pseudorandom strings of length  $n$ . Hence it was observed in [All89,All92] that if secure pseudorandom generators exist, then for all dense languages  $L$  in P/poly,  $\text{Kt}_L(n) = O(n^\epsilon)$  or  $\text{Kt}_L(n) = O(\log n)$  (depending on the security assumptions that were made about the pseudorandom generators).

We have seen that there is reason to be interested in the rate of growth that is possible for functions  $\text{Kt}_L$  when  $L$  is in P. One might also ask about  $\text{Kt}_L$  when  $L$  is chosen from some other complexity class. The following observation (essentially identical to an observation credited to Russo in [AR88]) is of interest in this regard.

**Theorem 3.** *The following are equivalent:*

- For every  $L$  in P,  $\text{Kt}_L(n) = O(\log n)$ .
- For every  $L$  in NP,  $\text{Kt}_L(n) = O(\log n)$ .
- For every  $L$  in  $\text{AC}^0$ ,  $\text{Kt}_L(n) = O(\log n)$ .

Similarly (by simply considering the (string,witness) pairs for a language in NP) it is easy to show:

**Theorem 4.** *The following are equivalent:*

- There is a language  $L$  in P and an  $\epsilon > 0$  such that for all large  $n$   $\text{Kt}_L(n)$  is defined and  $\text{Kt}_L(n) > n^\epsilon$ .
- There is a language  $L$  in NP and an  $\epsilon > 0$  such that for all large  $n$   $\text{Kt}_L(n)$  is defined and  $\text{Kt}_L(n) > n^\epsilon$ .
- There is a language  $L$  in  $\text{AC}^0$  and an  $\epsilon > 0$  such that for all large  $n$   $\text{Kt}_L(n)$  is defined and  $\text{Kt}_L(n) > n^\epsilon$ .

### 3 A Brief Discussion of Derandomization

One of the best examples of derandomization is provided by the lovely work of Impagliazzo and Wigderson:

**Theorem 5.** [IW97] *If there is a language  $A \in \text{E}$  and a constant  $\epsilon > 0$  such that, for all large  $n$ , there is no circuit of size  $2^{\epsilon n}$  accepting  $A^{=n}$ , then  $\text{BPP} = \text{P}$ .*

Alternate proofs of this theorem of [IW97] can be found in [STV01].

The hypothesis to Theorem 5 seems very likely to be true. Indeed, it seems possible that there are problems in smaller complexity classes than E (including perhaps SAT itself) that require exponential-size circuits. It also seems likely that E contains problems that are significantly harder than this hypothesis assumes. For instance, it seems likely that E requires circuits of this size, even if the circuits are allowed to have “oracle gates” for SAT [AK01]. It is shown in [KvM99] that this stronger hypothesis about the complexity of problems in E implies that  $AM = MA = NP$ . In [KvM99] the same conclusion  $AM=NP$  is shown to follow from a weaker assumption: namely that the “hard” set  $A$  is in  $NE \cap coNE$ . This hypothesis was subsequently weakened further by [MV99].

If one is willing to settle for a weaker conclusion than  $BPP=P$ , then it suffices to start with a much weaker assumption.

**Definition 3.** *Let  $t$  be a time bound. Define  $io-DTime(t(n))$  to be the class of languages  $L$  for which there exists a language  $A \in DTime(t(n))$  with the property that, for infinitely many  $n$ ,  $L^n = A^n$ .*

**Theorem 6.** [BFNW93] *If  $EXP$  is not in  $P/poly$ , then*

$$BPP \subseteq \bigcap_{\epsilon} io-DTime(2^{n^{\epsilon}}).$$

This is an example of “limited derandomization”; while it does not yield a polynomial-time algorithm for BPP problems, it does improve over the trivial exponential-time upper bound (at least for infinitely many input lengths). It is difficult to imagine that languages in BPP could be easy to recognize on some input lengths and difficult to recognize on other input lengths. Nonetheless, it is still not known if “io-Dtime” can be replaced by “Dtime” in the preceding theorem.

The conclusion of Theorem 6 implies that  $EXP$  is not contained in BPP – but it is not known if the ability to do limited derandomization implies that  $EXP$  is not in  $P/poly$ . That is, lower bounds are *sufficient* for limited derandomization, but lower bounds are not known to be *necessary* for derandomization. Stated another way, it would be interesting to know if the nonexistence of circuit lower bounds for problems in  $EXP$  (i.e.,  $EXP \subseteq P/poly$ ) implies that no derandomization is possible (i.e., that  $BPP = EXP$ ).

In contrast, it is shown by Impagliazzo, Kabanets, and Wigderson (in part credited by them to van Melkebeek) that lower bounds for NEXP are *necessary and sufficient* for limited derandomization of MA:

**Theorem 7.** [IKW01] *The following are equivalent:*

- $NEXP \not\subseteq P/poly$
- $NEXP \neq MA$
- $MA \subseteq \bigcap_{\epsilon} io(NTime(2^{n^{\epsilon}})/n^{\epsilon})$ .

Central to the approach taken in [IKW01] is the strategy of searching for solutions to NEXP-search problems by looking for “easy witnesses” first, where a string of length  $2^n$  is considered to be an “easy witness” if it is the truth table of a function with small circuit complexity.

This leads us to the following two questions:

**Question 1:** We already know that Kt complexity defines an optimal search strategy for NEXP search problems. How does this compare to the “easy witness” strategy of [IKW01]?

**Question 2:** Is the notion of “easy witnesses” also related to a variant of Kolmogorov complexity?

## 4 A New Definition of Time-Bounded Kolmogorov Complexity

To try to find an answer to these questions, we are led to a new definition of time-bounded Kolmogorov complexity. The definition below is a slight variant on a definition introduced by Antunes, Fortnow, and van Melkebeek in [AFvM01].

**Definition 4.** Let  $U$  be a universal Turing machine. Define  $\text{KT}(x)$  to be

$$\min\{|d| + t : \text{for all } i \leq |x|, U(d, i) = x_i \text{ in at most } t \text{ steps}\}.$$

(A note on notation: The capital “T” in KT calls attention to the fact that the time bound has much more influence than in Kt complexity.) In the preceding definition, note that the string  $d$  is “almost” a description of  $x$  in the usual sense; it is possible that  $U$  cannot determine the length of  $x$  from  $d$ . Of course, by adding  $O(\log n)$  to the length of  $d$ , we could require that  $U(d, i) = \perp$  for all  $i > |x|$  (thus obtaining a description of  $x$  in the usual sense). Logarithmic terms are insignificant for our applications, and thus we use the simpler definition.

Observe that the KT and Kt measures can be generalized to obtain  $\text{KT}^A$  and  $\text{Kt}^A$  for oracles  $A$ , by giving  $U$  access to an oracle. It turns out that this is useful in clarifying the relationship between Kt and KT complexity.

**Theorem 8 (Ronneburger).** [Ron01] Let  $A$  be complete for E under linear-time reductions. Then there is a  $k \in \mathbb{N}$  such that  $\text{Kt}(x)/k < \text{KT}^A(x) < k\text{Kt}(x)$ . In other words,  $\text{Kt} = \Theta(\text{KT}^A)$ .

At first glance, the definition of KT complexity seems to have quite a different flavor than Levin’s Kt complexity, especially because Levin’s definition uses the log of the running time, and KT complexity uses running time. However, the preceding theorem shows that one can view Kt complexity as merely a variation of KT complexity; Kt complexity is KT complexity relative to E.

Next, let us observe that KT complexity captures the essential character of the “easy witness” approach of [IKW01].

**Theorem 9.** Let  $x$  be a string of length  $2^n$  (which we can view as the truth table of a function  $f$  on inputs of length  $n$ ), and let  $A$  be any oracle. If  $\text{KT}^A(x) < m$  then there is an  $A$ -circuit of size  $O(m^3)$  computing  $f$ . Conversely, if there is an  $A$ -circuit of size  $m$  computing  $f$ , then  $\text{KT}^A(x) = O(m \log m)$ .

That is,  $\text{KT}(x)$  is an approximation to the circuit size required to compute the function given by the truth-table  $x$ .

It will be useful for us to define an analog to the function  $\text{Kt}_L$ .

**Definition 5.** Let  $L \subseteq \Sigma^*$ . Define  $\text{KT}_L(n)$  to be  $\min\{\text{KT}(x) : x \in L^n\}$ .

The following two theorems have the same easy proofs as the corresponding observations for  $\text{Kt}_L$ .

**Theorem 10.** *The following are equivalent:*

- For every  $L$  in  $\text{P}$ ,  $\text{KT}_L(n) = \log^{O(1)} n$ .
- For every  $L$  in  $\text{NP}$ ,  $\text{KT}_L(n) = \log^{O(1)} n$ .
- For every  $L$  in  $\text{AC}^0$ ,  $\text{KT}_L(n) = \log^{O(1)} n$ .

**Theorem 11.** *The following are equivalent:*

- There is a language  $L$  in  $\text{P}$  and an  $\epsilon > 0$  such that for all large  $n$   $\text{KT}_L(n)$  is defined and  $\text{KT}_L(n) > n^\epsilon$ .
- There is a language  $L$  in  $\text{NP}$  and an  $\epsilon > 0$  such that for all large  $n$   $\text{KT}_L(n)$  is defined and  $\text{KT}_L(n) > n^\epsilon$ .
- There is a language  $L$  in  $\text{AC}^0$  and an  $\epsilon > 0$  such that for all large  $n$   $\text{KT}_L(n)$  is defined and  $\text{KT}_L(n) > n^\epsilon$ .

#### 4.1 How Do KT and Kt Compare?

**Proposition 1.**  $\frac{1}{3}\text{Kt}(x) \leq \text{KT}(x) \leq 2^{1+\text{Kt}(x)}$ .

It is reasonable to conjecture that one of the two inequalities above is essentially tight. This leads us to formulate the two hypotheses below.

**Hypothesis A:**  $\text{KT}(x) = \text{Kt}(x)^{O(1)}$ .

**Hypothesis B:** There is some  $\epsilon > 0$  such that for all large  $n$  there is a string  $x \in \Sigma^n$  such that  $\text{KT}(x) > 2^{\epsilon \text{Kt}(x)}$ .

Hypothesis A says that the first inequality is essentially tight, and Hypothesis B says that the second inequality is essentially tight.

In the following section, we will see that each of these hypotheses has already been the object of a great deal of attention in the complexity theory community.

## 5 The Hypotheses are Familiar

First, let us consider Hypothesis A, which states that there is not much difference between  $\text{KT}$  and  $\text{Kt}$ . This is equivalent to a number of other statements, all of which seem highly unlikely.

**Theorem 12.** *The following are equivalent.*

1. Hypothesis A
2.  $\text{EXP} \subseteq \text{P/poly}$ .

3. For all P-printable sets  $L$ ,  $\text{KT}_L(n) = \log^{O(1)} n$ .
4. There exists a dense set  $L \in \text{P/poly}$  and some  $\epsilon > 0$  such that for all large  $n$ ,  $\text{Kt}_L(n)$  is defined and  $\text{Kt}_L(n) > n^\epsilon$ .

*Proof.* The equivalence (1.  $\iff$  2.) is straightforward. (Merely examine the KT complexity of prefixes of the characteristic sequence of a complete language for E.) Essentially the same observations suffice to show (3.  $\iff$  1.)

The proof of Theorem 6 (see [BFNW93]) actually establishes the implication (4.  $\implies$  2.) That is, the authors of [BFNW93] show that if  $\text{EXP} \not\subseteq \text{P/poly}$ , then the Nisan-Wigderson pseudorandom generator ([NW94]) can be used with seed length  $n^\epsilon$  to simulate any BPP algorithm. In fact, their analysis shows that the generator must output a string accepted by any small circuit that accepts many strings, which is precisely the negation of condition 4.

For the implication (2.  $\implies$  4.) we use an argument of [ISW99]. Assume that condition 4. fails to hold. That is, for every dense  $L \in \text{P/poly}$  and every  $\epsilon > 0$ , there are infinitely many  $x \in L$  such that  $\text{Kt}(x) < |x|^\epsilon$ . This yields a “hitting set generator;” i.e., a function  $G_\epsilon$  running in time  $2^{O(n^\epsilon)}$  that, on input  $n$ , outputs a list of strings such that any dense language in  $\text{P/poly}$  must accept one of the strings in the list. The complement of the range of  $G_\epsilon$  is dense, and is in E. If it were in  $\text{P/poly}$  it would contradict the existence of  $G_\epsilon$ .

The equivalence (3.  $\iff$  4.) is intriguing. It says that assuming that one class of sets has *small* Kolmogorov complexity is equivalent to another class of sets having *large* Kolmogorov complexity (using slightly different notions of time-bounded Kolmogorov complexity). This tension between high and low complexity is responsible for many of the most exciting aspects of derandomization.

Klivans and van Melkebeek show that the condition  $\text{EXP} \not\subseteq \text{P/poly}$  is sufficient to carry out limited derandomization of MA. If a stronger hypothesis is used, (i.e., one assumes that some P-printable set has large  $\text{KT}^{\text{SAT}}$  complexity instead of merely large KT complexity) then one obtains a derandomization of AM [KvM99].

Let us now turn our attention from Hypothesis A to Hypothesis B.

**Theorem 13.** *The following are equivalent:*

1. Hypothesis B
2. There exists a P-printable set  $L$  and an  $\epsilon > 0$  such that for all large  $n$ ,  $\text{KT}_L(n)$  is defined and  $\text{KT}_L(n) > n^\epsilon$ .
3. For all dense  $L$  in  $\text{P/poly}$ ,  $\text{Kt}_L(n) = O(\log n)$ .
4. There is a language  $A \in \text{E}$  and a constant  $\epsilon > 0$  such that, for all large  $n$ , there is no circuit of size  $2^{\epsilon n}$  accepting  $A^{=n}$ .
5. Efficient pseudorandom generators  $G$  exist, such that  $G : \Sigma^{k \log n} \rightarrow \Sigma^n$ .
6. Efficient hitting set generators exist.

That is, in particular, Hypothesis B is equivalent to the hypothesis of Theorem 5.

*Proof.* Items 4. through 6. in the list above have been observed before to be equivalent [For01,KRC00,ISW99]. The equivalence (1.  $\iff$  2.  $\iff$  4.) is straightforward and similar to the proof of Theorem 12. It remains only to consider condition 3.

Condition 3. is easily seen to be equivalent to the existence of a hitting set generator (i.e., a polynomial-time-computable function that, on input  $n$ , outputs a list  $S$  of strings of length  $n$  with the property that any circuit of size  $n$  accepting at least  $2^n/n$  elements of  $\Sigma^n$  must accept an element of  $S$ ). If we assume 3. is true, then the routine that lists all strings of length  $n$  having Kt complexity  $O(\log n)$  is a hitting set generator. Conversely, if we assume that hitting set generators exist, note that all strings produced by the generator have Kt complexity  $O(\log n)$ .

The preceding theorems gave equivalent ways to view the question of whether or not P-printable sets can have large KT-complexity. The following theorem shows that it is of interest to consider the KT-complexity of arbitrary sets in P.

**Theorem 14.** *The following are equivalent:*

1. For all  $L \in \text{P}$ ,  $\text{KT}_L(n) = \log^{O(1)} n$ .
2. All NEXP search problems are solvable in P/poly.
3.  $\text{NEXP} \subseteq \text{P/poly}$
4.  $\text{NEXP} = \text{MA}$
5.  $\text{MA} \not\subseteq \bigcap_{\epsilon} \text{io}(\text{NTime}(2^{n^\epsilon})/n^\epsilon)$ .

*Proof.* The equivalence (1.  $\iff$  2.) is straightforward. Equivalences (3.  $\iff$  4.  $\iff$  5.) and implication (3.  $\implies$  2.) were established in [IKW01]. The remaining implication (2.  $\implies$  3.) is trivial.

## 6 The Question of Density

It is obvious that there are sets  $L$  in P/poly (and even in non-uniform  $\text{AC}^0$ ) with high  $\text{Kt}_L$  complexity (and hence with high  $\text{KT}_L$  complexity), since for each length  $n$  one can select a string  $x_n$  of length  $n$  with high Kt-complexity, and build a circuit that accepts exactly  $x_n$ . However, as observed earlier, dense languages  $L$  in P/poly must have low  $\text{Kt}_L$  complexity, or else secure pseudorandom generators do not exist.

Theorems 3, 4, 10, and 11 show that the Kolmogorov complexity of sets in P and  $\text{AC}^0$  are closely related. It is natural to wonder if the Kolmogorov complexity of dense sets in P/poly and non-uniform  $\text{AC}^0$  are also related. No such relationship is known, although dense sets in  $\text{AC}^0$  provide one of the few examples where we are able to say something nontrivial about Kt and KT complexity.

It was essentially observed by [Zim97] that the derandomization results of Nisan and Wigderson [NW94] can be used to show that, for dense  $L$  in  $\text{AC}^0$ ,  $\text{Kt}_L(n) = \log^{O(1)} n$ . Closer examination of the Nisan-Wigderson generator shows that an even stronger conclusion holds:

**Theorem 15.** *Let  $L$  be a dense language in  $AC^0$ . Then  $Kt_L(n) = \log^{O(1)} n$ .*

For even slightly larger classes (such as  $AC^0[2]$ ) no nontrivial bound on the  $Kt_L$  or  $Kt_L$  complexity is known.

It is natural to wonder if the bound of  $\log^{O(1)} n$  in the preceding theorem can be improved to  $O(\log n)$ . The results of Agrawal [Agr01b] can be used to show that this question is equivalent to a question in circuit complexity:

**Theorem 16.** *The following are equivalent.*

- For all dense  $L$  in non-uniform  $AC^0$ ,  $Kt_L(n) = O(\log n)$ .
- For some  $\epsilon > 0$ ,  $E \not\subseteq AC^0(2^{\epsilon n})$ .

## 7 Natural Proofs, Lower Bounds, and Kolmogorov Complexity

We have seen that there are good reasons to believe that dense sets in P/poly must have low Kt complexity, but we have not discussed any consequences of the existence of dense sets in P/poly with large KT complexity.

It turns out that this is exactly the question that was raised by Razborov and Rudich when they developed their framework of “natural proofs” to explain why certain approaches to proving circuit lower bounds appear to be doomed to failure [RR97]. Let us recall some of the definitions of Razborov and Rudich.

**Definition 6.** [RR97] *A Combinatorial Property of Boolean functions is a subset of  $\{\Sigma^{2^n} : n \in \mathbb{N}\}$ ; that is, it is a set of truth tables.*

*A combinatorial property  $C$  is useful against P/poly if any language  $L$  in P/poly has the property that  $\{n : \text{the truth table for } L^{=n} \text{ lies in } C\}$  is finite.*

Razborov and Rudich also consider a “largeness” condition; we shall discuss that later.

Note that, by Theorem 9, if  $Kt(x)$  is large, then there is some  $i < |x|$  such that  $x0^i$  is the truth table of a function requiring large circuits. From this simple observation, we can see that a language  $L$  where  $\forall k \forall^\infty n Kt_L(n) \geq \log^k n$  is the same thing as a combinatorial property useful against P/poly.

Impagliazzo, Kabanets, and Wigderson observed in [IKW01] that if there is a combinatorial property in NP that is useful against P/poly, then  $NEXP \not\subseteq P/poly$ . They then asked if the existence of a combinatorial property in P that is useful against P/poly implies that  $EXP \not\subseteq P/poly$ . However, as in the proof of Theorem 10, we can see that there is a combinatorial property in NP that is useful against P/poly if and only if such a property exists in  $AC^0$ .

The other crucial ingredient that a combinatorial property needs in order to be “natural” (in the framework of Razborov and Rudich) is that it needs to be dense. (That is, it must contain at least  $2^N/N^k$  strings, for all  $N$  of the form  $2^n$ . All theorems in this paper involving “dense” sets carry over if we change our density requirement from  $2^n/n$  to  $2^n/n^{O(1)}$ .)

Razborov and Rudich showed that, under strong (but widely-believed) assumptions about the existence of secure pseudorandom generators, there is no dense combinatorial property in P/poly that is useful against P/poly. Hence, under the hypothesis of [RR97], for all dense  $L \in \text{P/poly}$ ,  $\text{KT}_L(n) = \log^{O(1)} n$ .

Note that there are very close connections between the notion of Natural Proofs and certain aspects of resource-bounded measure theory [RSC95,Lut92]. There is not space here to elaborate on these connections. However, this suggests that there may be interesting implications between time-bounded Kolmogorov complexity and resource-bounded measure that deserve to be explored.

## 8 Random Strings

The canonical dense sets with large Kolmogorov complexity are the sets containing *all* of the random strings. For our purposes, it will suffice to consider the following two sets:

- $R_{\text{KT}}$  is defined to be  $\{x \mid \text{KT}(x) \geq |x|/2\}$ .
- $R_{\text{Kt}}$  is defined to be  $\{x \mid \text{Kt}(x) \geq |x|/2\}$ .

Similar (but not identical) notions of time-bounded randomness have been considered before [BT01,Ko91,Har83,KC00].

$R_{\text{KT}}$  is in coNP, and  $R_{\text{Kt}}$  is in E. It seems natural to conjecture that these upper bounds are essentially optimal. However, there are significant obstacles to showing that there are no smaller complexity classes containing these languages. Most significantly, there is reason to believe that these languages are *not complete* for coNP and E, respectively. For some notions of completeness, it is even possible to prove that this is the case.

**Theorem 17.**  *$R_{\text{Kt}}$  is not hard for E under polynomial-time many-one reductions.*

*Proof.* We provide only a sketch. Let  $T$  be a subset of  $0^*$  that is in E but not in P. Let  $f$  be any polynomial-time function. Note that  $\text{Kt}(f(0^n)) = O(\log n)$ . Thus  $f(0^n)$  is not in  $R_{\text{Kt}}$  unless  $f(0^n)$  is *very* short – in which case membership in  $R_{\text{Kt}}$  can be determined in polynomial time.

Essentially the same proof also shows that  $R_{\text{Kt}}$  is not complete under polynomial-time truth-table reductions.

Buhrman and Mayordomo show that a related (but not identical) language is not hard for E even under polynomial-time Turing reductions (and in fact they obtain even stronger conclusions relating to resource-bounded measure) [BM97]. However, their argument does not seem to extend to  $R_{\text{Kt}}$ .

By Theorem 15, neither  $R_{\text{Kt}}$  nor  $R_{\text{KT}}$  is in (non-uniform)  $\text{AC}^0$ . Somewhat surprisingly, as the deadline for submission of this paper approaches I am unable to find or construct any other lower bounds on the complexity either of these problems (other than conditional lower bounds, such as the argument in [RR97]

showing that these sets are not in  $P/\text{poly}$  if strong pseudorandom generators exist).

Ko presents oracles relative to which a set closely related to  $R_{KT}$  is not complete for  $\text{coNP}$ . The recent proof by Agrawal, showing that all sets complete for  $\text{NP}$  under  $\text{AC}^0$  many-one reductions are  $\text{AC}^0$ -isomorphic [Agr01a] can be used to show that  $R_{Kt}$  is not hard even for some very small subclasses of  $P$ . (In addition, it provides conditions where  $R_{KT}$  also fails to be hard).

**Theorem 18.**  *$R_{Kt}$  is not hard for  $\text{TC}^0$  under  $\text{AC}^0$ -many-one reductions. Also, if SAT has circuits of size  $2^{n^{o(1)}}$ , then  $R_{KT}$  is not hard for  $\text{TC}^0$  under  $\text{AC}^0$ -many-one reductions.*

*Proof.* Again, we provide only a sketch. Agrawal shows in [Agr01a] that any set hard for  $\text{TC}^0$  under  $\text{AC}^0$  reductions is hard under length-increasing  $\text{NC}^0$  reductions where the connections in the  $\text{NC}^0$  circuits are computable in  $\text{AC}^0$ . Thus, in particular, the output of the reduction has small  $\text{Kt}$  complexity and cannot lie in  $R_{Kt}$ . For  $\text{KT}$  complexity, we require an additional assumption. Under the assumption that SAT has small circuits, there is a small circuit that takes  $i$  as input, and produces an  $\text{NC}^0$  circuit for the  $i$ -th bit of the reduction as output. Now one can show that the output of the reduction must have small  $\text{KT}$  complexity, and hence foil the reduction.

For instance, if  $P=\text{NP}$ , then  $R_{Kt} \notin P$  (by Theorem 13) and  $R_{KT} \in P$ , but  $R_{KT}$  is not hard for  $\text{TC}^0$  under  $\text{AC}^0$  many-one reductions.

Cai and Kabanets study a question of whether a set closely related to  $R_{KT}$  is in  $P$  or is  $\text{NP}$ -complete. For instance, if  $R_{KT}$  is complete under length-increasing polynomial-time reductions, then there is a  $P$ -printable set  $L$  such that  $\text{Kt}_L$  grows very quickly, and hence one obtains all of the consequences listed in Theorem 13.

## 9 Conclusions

Kolmogorov complexity has been very useful in attacking a wide variety of problems in computer science and mathematics. It has not found much application (yet) in the field of derandomization. (A notable exception is the work of Trevisan [Tre01].) It is hoped that the definitions provided here will be useful in formulating new lines of attack on these problems.

In particular, it seems quite likely to me that better bounds can be proved on the complexity of  $R_{KT}$ . It should be possible to prove sublinear bounds on the growth of  $\text{Kt}_L$  for dense sets  $L$  in  $\text{AC}^0[2]$ . I feel sure that there are new and interesting implications waiting to be discovered involving the  $\text{KT}_L$  and  $\text{Kt}_L$  complexity of various classes of sets. Since these notions have tight connections to the theory of natural proofs (and hence to resource-bounded measure), I think it is possible that investigation of these questions will have application not only to the study of derandomization, but throughout complexity theory.

## Acknowledgments

This work was supported by National Science Foundation grant number CCR-0104823. I thank Russell Impagliazzo, Lance Fortnow, Detlef Ronneburger, Valentine Kabanets, and Dieter van Melkebeek for helpful comments and for pointing out errors in an earlier draft of this paper. I also thank Detlef Ronneburger for allowing me to report on our ongoing work here.

## References

- [AFvM01] Luis Antunes, Lance Fortnow, and Dieter van Melkebeek. Computational depth. In *IEEE Conference on Computational Complexity*, pages 266–273, 2001.
- [Agr01a] M. Agrawal. The first-order isomorphism theorem. In *Proceedings 21st Foundations of Software Technology and Theoretical Computer Science Conference (FST&TCS)*, Lecture Notes in Computer Science. Springer, 2001.
- [Agr01b] M. Agrawal. Hard sets and pseudo-random generators for constant depth circuits. In *Proceedings 21st Foundations of Software Technology and Theoretical Computer Science Conference (FST&TCS)*, Lecture Notes in Computer Science. Springer, 2001.
- [AK01] V. Arvind and Johannes Köbler. On pseudorandomness and resource-bounded measure. *Theoretical Computer Science*, 255:205–221, 2001.
- [All89] Eric Allender. Some consequences of the existence of pseudorandom generators. *Journal of Computer and System Sciences*, 39:101–124, 1989.
- [All92] Eric Allender. Applications of time-bounded kolmogorov complexity in complexity theory. In *Osamu Watanabe (Ed.), Kolmogorov Complexity and Computational Complexity*, pages 4–22. Springer-Verlag, 1992.
- [AR88] E. Allender and R. Rubinfeld. P-printable sets. *SIAM J. Comput.*, 17:1193–1202, 1988.
- [AW90] Eric Allender and Osamu Watanabe. Kolmogorov complexity and degrees of tally sets. *Information and Computation*, 86:160–178, 1990.
- [BFL01] H. Buhrman, L. Fortnow, and S. Laplante. Resource-bounded Kolmogorov complexity revisited. *SIAM Journal on Computing*, 2001. to appear; a preliminary version appeared in STACS 1997.
- [BFNW93] László Babai, Lance Fortnow, Noam Nisan, and Avi Wigderson. BPP has subexponential time simulations unless EXPTIME has publishable proofs. *Computational Complexity*, 3:307–318, 1993.
- [BM88] L. Babai and S. Moran. Arthur-Merlin games: A randomized proof system, and a hierarchy of complexity classes. *Journal of Computer and System Sciences*, 36:254–276, 1988.
- [BM97] Harry Buhrman and Elvira Mayordomo. An excursion to the Kolmogorov random strings. *Journal of Computer and System Sciences*, 54:393–399, 1997.
- [BT01] Harry Buhrman and Leen Torenvliet. Randomness is hard. *SIAM Journal on Computing*, 30:1485–1501, 2001.
- [DK00] D.-Z. Du and K.-I. Ko. *Theory of Computational Complexity*. Wiley-Interscience, New York, 2000.

- [For01] Lance Fortnow. Comparing notions of full derandomization. In *IEEE Conference on Computational Complexity*, pages 28–34, 2001.
- [Har83] Juris Hartmanis. Generalized Kolmogorov complexity and the structure of feasible computations (preliminary report). In *IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 439–445, 1983.
- [Hås88] J. Håstad. *Computational Limitations of Small Depth Circuits*. MIT Press, Cambridge, 1988.
- [HY84] J. Hartmanis and Y. Yesha. Computation times of NP sets of different densities. *Theoretical Computer Science*, 34:17–32, 1984.
- [IKW01] R. Impagliazzo, V. Kabanets, and A. Wigderson. In search of an easy witness: Exponential time vs. probabilistic polynomial time. In *IEEE Conference on Computational Complexity*, pages 2–12, 2001.
- [Imm88] N. Immerman. Nondeterministic space is closed under complement. *SIAM Journal on Computing*, 17:935–938, 1988.
- [ISW99] R. Impagliazzo, R. Shaltiel, and A. Wigderson. Near-optimal conversion of hardness into pseudo-randomness. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 181–190, 1999.
- [IT89] R. Impagliazzo and G. Tardos. Decision versus search problems in super-polynomial time. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 222–227, 1989.
- [IW97] R. Impagliazzo and A. Wigderson.  $P = BPP$  if  $E$  requires exponential circuits: Derandomizing the XOR lemma. In *ACM Symposium on Theory of Computing (STOC)*, pages 220–229, 1997.
- [KC00] Valentine Kabanets and Jin-Yi Cai. Circuit minimization problem. In *ACM Symposium on Theory of Computing (STOC)*, pages 73–79, 2000.
- [Ko91] Ker-I Ko. On the complexity of learning minimum time-bounded Turing machines. *SIAM Journal on Computing*, 20:962–986, 1991.
- [KRC00] Valentine Kabanets, Charles Rackoff, and Stephen Cook. Efficiently approximable real-valued functions. In *Electronic Colloquium on Computational Complexity (ECCC) Report TR00-034*, 2000. ([www.eccc.uni-trier.de/eccc](http://www.eccc.uni-trier.de/eccc)).
- [KvM99] A. Klivans and D. van Melkebeek. Graph nonisomorphism has subexponential size proofs unless the polynomial-time hierarchy collapses. In *ACM Symposium on Theory of Computing (STOC)*, pages 659–667, 1999.
- [Lev84] Leonid A. Levin. Randomness conservation inequalities; information and independence in mathematical theories. *Information and Control*, 61:15–37, 1984.
- [LFKN92] Carsten Lund, Lance Fortnow, Howard Karloff, and Noam Nisan. Algebraic methods for interactive proof systems. *Journal of the ACM*, 39:859–868, 1992.
- [Lut92] Jack H. Lutz. Almost everywhere high nonuniform complexity. *Journal of Computer and System Sciences*, 44:220–258, 1992.
- [MV99] Peter Bro Miltersen and N. V. Vinodchandran. Derandomizing Arthur-Merlin games using hitting sets. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 71–80, 1999.
- [NW94] N. Nisan and A. Wigderson. Hardness vs. randomness. *Journal of Computer and System Sciences*, 49:149–167, 1994.
- [Pap94] C. Papadimitriou. *Computational Complexity*. Addison-Wesley, New York, 1994.
- [Ron01] Detlef Ronneburger. Personal communication. 2001.
- [RR97] Alexander A. Razborov and Steven Rudich. Natural proofs. *Journal of Computer and System Sciences*, 55:24–35, 1997.

- [RSC95] K. W. Regan, D. Sivakumar, and Jin-Yi Cai. Pseudorandom generators, measure theory, and natural proofs. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 26–35, 1995.
- [Sha92] A. Shamir.  $IP = PSPACE$ . *Journal of the ACM*, 39:869–877, 1992.
- [STV01] M. Sudan, L. Trevisan, and S. Vadhan. Pseudorandom generators without the XOR lemma. *Journal of Computer and System Sciences*, 62:236–266, 2001.
- [Sze88] R. Szelepcsényi. The method of forced enumeration for nondeterministic automata. *Acta Informatica*, 26:279–284, 1988.
- [Tre01] Luca Trevisan. Construction of extractors using pseudo-random generators. *Journal of the ACM*, 2001. to appear; a preliminary version appeared in STOC 1999.
- [Vol99] H. Vollmer. *Introduction to Circuit Complexity*. Springer-Verlag, 1999.
- [Zim97] Marius Zimand. Large sets in  $AC^0$  have many strings with low Kolmogorov complexity. *Information Processing Letters*, 62:165–170, 1997.