

# Measure on Small Complexity Classes, with Applications for BPP

Eric Allender\*

Department of Computer Science  
Rutgers University  
New Brunswick, NJ 08903  
allender@cs.rutgers.edu

Martin Strauss\*

Department of Mathematics  
Rutgers University  
New Brunswick, NJ 08903  
mstrauss@math.rutgers.edu

## Abstract

*We present a notion of resource-bounded measure for P and other subexponential-time classes. This generalization is based on Lutz’s notion of measure, but overcomes the limitations that cause Lutz’s definitions to apply only to classes at least as large as E. We present many of the basic properties of this measure, and use it to explore the class of sets that are hard for BPP.*

*Bennett and Gill showed that almost all sets are hard for BPP; Lutz improved this from Lebesgue measure to measure on ESPACE. We use our measure to improve this still further, showing that for all  $\epsilon > 0$ , almost every set in  $E_\epsilon$  is hard for BPP, where  $E_\epsilon = \bigcup_{\delta < \epsilon} \text{DTIME}(2^{n^\delta})$ , which is the best that can be achieved without showing that BPP is properly contained in E. A number of related results are also obtained in this way.*

## 1 Introduction

Resource-bounded measure theory was introduced in [L90a] and has proved to be a useful tool in complexity theory. Among the reasons for studying resource-bounded measure, we mention the following:

- The probabilistic method can be applied only if there is a meaningful notion of probability on the underlying space. To show that a set in E having some property  $Q$  exists, it is sufficient to show that the class of sets that do

*not* have property  $Q$  has measure zero in E. (This sometimes presents itself as an easier task than explicitly building a set in E having property  $Q$ .) For example, Lutz and Mayor-domo [LM] were able to use this version of the probabilistic method to prove stronger results concerning the density of hard languages than had been proved previously.

- Certain interesting notions of intractability can be formulated only in the context of resource-bounded measure. Lutz and Juedes [JL] studied “weakly complete” sets in E, which are sets in E that are hard for more than a measure-zero subset of E.
- Many separation results and hierarchy theorems in complexity theory have stronger versions provable in the context of resource-bounded measure. For example, the  $\leq_m^p$ -complete sets are a measure-zero subset of E [JL], and  $\text{DTIME}(2^{cn})$  is a measure-zero subset of E [L92]. However, for many other separations, no corresponding measure result is known. Often, as in [JL], when a measure-theoretic strengthening of a separation result is possible, the proof gives enlightening additional information concerning the complexity classes involved.
- Unproven but plausible hypotheses such as “ $P \neq NP$ ” or the stronger hypothesis “the polynomial hierarchy is infinite” provide useful information concerning complexity-theoretic propositions. (That is, “ $X \Rightarrow$  the polynomial hierarchy collapses” is taken as

---

\*Research supported by NSF grant CCR-9204874.

evidence for  $\neg X$ .) Lutz [L93a] argues that the hypothesis “NP is not a measure zero subset of  $\text{DTIME}(2^{n^{O(1)}})$ ” (a stronger hypothesis than  $P \neq \text{NP}$ ) is plausible and offers a great deal of explanatory power.

Unfortunately, the formulation of resource-bounded measure studied by Lutz and others applies only to complexity classes at least as large as E, which greatly limits the range of questions that can be explored in this framework. The goal of the work presented here is to remedy this situation by providing a meaningful notion of measure for P and other subexponential time classes, and demonstrate its usefulness.

Section 2 presents Lutz’s definition of resource-bounded measure, explains why it does not extend in any obvious way to provide a measure on smaller classes, and then shows how these obstacles were overcome to provide a measure on P and other classes. This section concludes with a discussion of how natural and robust our definition of measure is, and a comparison of it to the work of Mayordomo [M, M2], where a measure on PSPACE is defined.

Section 3 presents our proof that sets that are hard for BPP are abundant in  $E_\epsilon$  for every  $\epsilon > 0$ , as well as related results for PSPACE.

Section 4 presents our results concerning pseudorandom sources for BPP.

Section 5 includes a discussion of whether the notion of measure introduced in this paper relativizes in a meaningful way. (That is, we discuss the question of whether nonrelativizing results might be obtained using this notion of measure.)

## 2 Defining Resource-Bounded Measure

### 2.1 Background Concerning Measure

In order to present our new definitions of resource-bounded measure and explain the obstacles to extending Lutz’s definitions to subexponential complexity classes, it is necessary to

sketch Lutz’s formulation. For more details, consult [L92].

The intuition behind Lutz’s formulation of resource-bounded measure is that a measure-zero set is a set such that, for all  $k$ , the set can be covered by a collection of intervals whose sizes sum to  $2^{-k}$ , where the “intervals” are easy to compute in some sense.

To make this more precise, fix an enumeration  $\{s_i\}_{i=0}^\infty$  of all words, and let  $\text{pos}(x)$  denote the position of  $x$  in this ordering. A language  $L$  is identified with its characteristic sequence: an infinite sequence  $\omega = \langle a_i \rangle$  with  $a_i = 1$  iff  $s_i \in L$ . Let  $w[i..j]$  denote the string of the  $i^{\text{th}}$  through  $j^{\text{th}}$  bits of  $w$ . Let  $y \sqsubseteq w$  indicate that  $y$  is an initial substring of  $w$ .

The sets we will try to measure are sets of languages (i.e., sets of sequences), and the analogs of “intervals” in this context are the sets of the form  $C_x = \{\omega \sqsupseteq x\}$ , which is called the “cylinder at  $x$ ”. (Thus  $C_x$  is the set of languages with the  $|x|$  initial words according to  $x$  and the other words arbitrary).

The next few definitions describe what it means in this context for a set to be covered, for all  $k$ , by a collection of intervals whose sizes sum to  $2^{-k}$ .

**Definition 1** *A density-system is a function  $d_{i_1, \dots, i_r} \{0, 1\}^* \rightarrow [0, \infty)$ , where –*

1. *The subscripts are natural numbers.*
2. *The argument is considered a partial characteristic sequence of a language.*
3.  $d_{(\cdot)}(w) = \frac{d_{(\cdot)}(w0) + d_{(\cdot)}(w1)}{2}$ .

(For Lutz’s measure, this definition is equivalent to the definition in [L92]; see also Subsection 2.5.) A density-system is called an  $n$ -DS according to the number of subscripts. A 0-DS is a density function. (Note there is no mention here of resource bounds.)

A density function *covers*  $X$  if  $X \subseteq \bigcup_{\{w \mid d(w) \geq 1\}} C_w$ . A *null cover* of  $X$  is a 1-DS such that for all  $k$ ,  $d_k$  covers  $X$  and  $d_k(\lambda) \leq 2^{-k}$ . A

set of languages has Lebesgue-measure zero iff it has a null cover. Of course, all recursive complexity classes are countable and thus have Lebesgue-measure zero. Thus it was necessary for Lutz to restrict the class of density systems in order to define measures on complexity classes.

In defining notions of measure<sup>1</sup> for E, ESPACE, and other large complexity classes, Lutz defines a  $\Delta$ -measure for various classes of functions  $\Delta$ . Roughly, the sets with  $\Delta$ -measure zero are the sets that have a null cover that can be approximated very closely by a function in  $\Delta$ . If  $\Delta$  is chosen to be the polynomial-time computable functions, Lutz obtains a measure on E; if  $\Delta$  is chosen to be the functions computable in space  $2^{\log^{O(1)} n}$ , then he obtains a measure on DSPACE( $2^{n^{O(1)}}$ ), and in general functions computed in space or time  $\{f(\log n) : f \in \mathcal{C}\}$  for a sufficiently nice class  $\mathcal{C}$  of functions yield a measure on space or time  $\{f(n) : f \in \mathcal{C}\}$ .

In order to justify his claim that the notion he defines does, in fact, constitute a reasonable notion of measure on a complexity class  $X$ , Lutz [L92] notes that if  $\Delta$  is chosen to be the class of *all* functions, then his definitions are equivalent to Lebesgue measure, and he also formulates three “measure axioms” that his system satisfies:

**M 1** *Easy unions of null sets are null.*

**M 2** *Singleton sets of easy languages are null.*

**M 3**  *$X$  itself is not null.*

Our goal in defining a measure on small complexity classes is to formulate a system that is as simple and natural as possible, while still satisfying these axioms.

When faced with the task of defining measure on classes smaller than E, it is natural to try to

---

<sup>1</sup>The classes of sets that we are interested in measuring are complexity classes, which are closed under finite variants and thus have measure zero or one if they are measurable at all. For this reason, and for the reason that an extra layer of complication is involved in making the notion of measure precise for sets of nonzero measure, we follow Lutz’s lead in defining only what it means for a set to have measure zero or one.

modify Lutz’s definitions, merely using smaller resource bounds  $\mathcal{C}$ . For instance, to define a measure on P, one would consider density functions computed in DTIME( $\log^{O(1)} n$ ). This fails to work, because

- The usual convention (see, e.g., [AJ]) for having sublinear-time Turing machines compute some function  $f$  is to have them recognize the language  $\{x, i, b : \text{bit } i \text{ of } f(x) \text{ is } b\}$ . However it is easy to see that the class of functions computed in this way by, for instance, polylog-time-bounded machines, is not closed under addition and subtraction, which seems to be necessary for many constructions. Similar problems arise when one uses the usual binary notation or scientific notation for the numeric values of the density functions. Our solution is to have the run time bound the length of the output, and to express numbers as the difference of two formal sums of powers of two, which allows us to perform the necessary arithmetic operations in the restricted time available.
- Using this representation for numeric values, one obtains a system that quite possibly *does* define a measure on P. It unfortunately seems quite difficult to verify that P itself does not have measure zero (which would violate Measure Axiom 3). This is discussed in more detail in [AS], but the central problem lies with an observation made previously in [M]: the binary sequences that are constructible in DTIME( $\log^{O(1)} n$ ) correspond *not* to sets in P, but rather to “word-decreasing self-reducible” sets, some of which are hard for E [Ba]. This motivates our notion of *limiting the dependency set size*, which allows us to obtain subexponential bounds on the complexity.

## 2.2 Formal Definitions of the Measure

The preceding paragraphs motivate some detailed definitions of a class of functions computed by sublinear-time Turing machines. In order for sublinear-time machines to perform interesting

computation, we follow the usual convention of providing these machines with random access to their input; that is, the machines have an “address tape,” and if  $i$  is written in binary on the address tape, then the machine can in unit time move its read/write head to bit position  $i$  of the input. Among other things, such machines can, in logarithmic time, compute the length of their input ([Bu]). In order to avoid uninteresting technicalities regarding encoding of pairing functions, we adopt the convention that a machine computing a  $k$ -ary function is provided with  $k$  input tapes (with an “address tape” for each input tape). The running time of such a machine must be polylogarithmic in  $m$ , where  $m$  is the sum of the lengths of its  $k$  inputs. (Note that by choosing a suitable encoding, such a machine can be simulated by a machine with a single input tape.) The machines we consider will write their output on a write-only output tape; thus the output is restricted to be of length bounded by the running time.

Given a machine  $M$  and natural number  $n$ , define a *dependency set*  $G_{M,n} \subseteq \{0, \dots, n\}$  to be a set such that for each  $i \in G_{M,n} \cup \{n\}$ , and each word  $w$  of length  $n$ ,  $M$  can compute  $M(w[0..i])$  querying only input bits in  $G_{M,n} \cap \{0, 1, 2, \dots, i\}$ . Note that for all  $M$  and  $n$ , there is a unique minimal dependency set for  $M$  and  $n$ , which can easily be computed by expanding the tree of queries that one obtains by assuming both possible values for each queried bit. In what follows, we let  $G_{M,n}$  denote this minimal dependency set. Given a function  $f$  computed by machine  $M$ , we may abuse notation and speak of  $G_{f,n}$  instead of  $G_{M,n}$ , where this causes no confusion. Note our convention about paired inputs to  $M$  requires that if  $M$  computes a subscripted function  $d_{k,r}$ ,  $G$  gets a cadre of subscripts matching  $d$ 's:  $G_{M,|w|,k,r}$  for  $d_{k,r}(w)$ . Often in practice  $G_{M,n,k,r}$  is independent of  $k$  and  $r$ ; in that case these subscripts will be suppressed.

Given a complexity class  $\mathcal{C}$  of the form  $\text{DTIME}(\mathcal{F})$  (where we will always assume that  $\mathcal{F}$  is a set of time-constructible functions such that  $f(n) \in \mathcal{F} \Rightarrow (f(n))^2 \in \mathcal{F}$ ), let  $\Delta(\mathcal{C})$  be the class of functions computed by Turing ma-

chines running in time  $f(\log n)$  for some  $f \in \mathcal{F}$ , and let  $\mathcal{C}$  be the class of functions  $d_{i_1, \dots, i_l}(w)$  computed by machines whose runtime and dependency set size are both bounded by functions of the form  $f(\log(i_1 + \dots + i_l + |w|))$  for some  $f \in \mathcal{F}$ .

Note that if the functions in  $\mathcal{F}$  are at least exponential, then  $\Delta(\mathcal{C}) = \mathcal{C}$ , and hence our definitions coincide with those of [L92] for complexity classes at least as large as  $E$ .<sup>2</sup>

If  $f$  is a function in  $\mathcal{C}$ , where  $f : \{0, 1\}^* \rightarrow \{0, 1\}$ , then  $f$  defines a *constructor*  $\delta$ , where  $\delta(w) = wf(w)$ . A constructor specifies the sequence that is the limit as  $j \rightarrow \infty$  of  $\delta^j(\lambda)$ . This gives rise to the class  $R(\mathcal{C})$ , which is the class of languages whose characteristic sequences are given by some constructor in  $\mathcal{C}$ . The definitions presented above were designed to maintain the following important property of the system developed in [L92]:

**Proposition 1**  $R(\mathcal{C}) = \mathcal{C}$ .

**Proof.** Any language in  $\mathcal{C}$  has a trivial constructor that ignores all but the length of its input, so we have  $\mathcal{C} \subseteq R(\mathcal{C})$ .

On the other hand, given such a constructor  $\delta$ , we can decide if  $x \in R(\delta)$  as follows: Start running  $\delta$  on  $w = \delta^{\text{pos}(x)-1}(\lambda)$ . When  $\delta$  needs bit  $i$  of  $w$ , recursively call  $\delta$  on  $w[0..i \leftrightarrow 1]$  to generate that bit. Unwinding this recursion effectively generates the dependency set  $G_{\delta,|w|}$ , which by hypothesis has size at most  $f(\log |w|) \approx f(|x|)$ . For

<sup>2</sup>One technicality that needs to be addressed concerns the manner in which machines determine the length of their input. If one were to use the usual binary search approach to determine the length of the input, then the dependency sets cannot be kept small while allowing any sort of interesting computation. Our approach to this problem is to have the length of the input be given explicitly as an argument to the machine. We follow the (useful) convention of [L92] that “subscripts” to density functions are presented in unary; however the reader should note that providing the length of a unary argument obviates any need to provide the argument itself. We continue to determine the complexity as if the subscripts were presented in unary. Since all but one of the inputs to the machine are in unary, the dependency set is only of interest for this one non-unary input. However, the dependency set *size* is allowed to depend on the length of the unary inputs.

each string  $y$  in the dependency set, we require at most time  $f(|y|)$ , for total computation time about  $(f(|x|))^2$ , which is in  $\mathcal{F}$  by the assumed closure properties. ■

We now make precise the notion of a density system (DS) being easy to compute. A  $(\mathcal{C})$ -computation of an  $n$ -DS  $d_{i_1, \dots, i_n}$  is an  $(n+1)$ -subscripted function  $\hat{d}_{i_1, \dots, i_n, r}$  satisfying

- $\hat{d}_{i_1, \dots, i_n, r}(w)$  is computable in  $(\mathcal{C})$ , (and numeric output is represented as a pair  $(a, b)$  representing the dyadic rational  $a \Leftrightarrow b$ , where each of  $a$  and  $b$  are represented as a formal sum of powers of 2).
- $|d_{i_1, \dots, i_n}(w) \Leftrightarrow \hat{d}_{i_1, \dots, i_n, r}(w)| \leq 2^{-r}$ .

Note that a  $(\mathcal{C})$ -computation does not have to satisfy the average law (item 3 of Definition 1), though it has to be close, depending on  $r$ .

For any complexity class  $\mathcal{C}$ , we say that a set  $X$  is  $(\mathcal{C})$ -null if there is a  $(\mathcal{C})$ -computation of a 1-DS  $d_k(w)$  covering  $X$  such that for all  $k$ ,  $d_k(\lambda) \leq 2^{-k}$ . In this case we write<sup>3</sup>  $\mu_{\Gamma(\mathcal{C})}(X) = 0$ .

If a class  $X$  has nonzero measure, we can talk about another set  $Y$  having “measure 0 in  $X$ ,” and we write  $\mu_{\Gamma(\mathcal{C})}(Y|X) = 0$ , if  $\mu_{\Gamma(\mathcal{C})}(Y \cap X) = 0$  or “measure 1 in  $X$ ,”  $\mu_{\Gamma(\mathcal{C})}(Y|X) = 1$ , if  $\mu_{\Gamma(\mathcal{C})}(X \setminus Y) = 0$ .

### 2.3 The Measure Axioms Are Satisfied

A set  $X$  is a  $\mathcal{C}$ -union of  $(\mathcal{C})$ -null sets if  $X = \bigcup_{j=0}^{\infty} X_j$ , and there is a 2-DS  $d_{j,k}$  so that  $d_{j,k}$  covers  $X_j$  with value  $2^{-k}$ , and  $d_{j,k}$  has a  $(\mathcal{C})$ -computation  $\hat{d}_{j,k,r}$ .

**Theorem 2** For each  $L \in \mathcal{C}$ , the singleton set  $\{L\}$  is  $(\mathcal{C})$ -null.

**Theorem 3** If  $X$  is a  $\mathcal{C}$ -union of  $(\mathcal{C})$ -null sets, then  $X$  is  $(\mathcal{C})$ -null.

<sup>3</sup>This is a slight change of notation from that used by Lutz and others. For example, if  $\mathcal{C}$  is taken to be  $\mathbf{E}$ , then  $\Delta(\mathcal{C}) = \mathbf{E}$ ,  $(\mathcal{C})$  is the class of functions computed in polynomial time, and if  $X$  is a measure-zero subset of  $\mathbf{E}$ , we would express that as  $\mu_{\Gamma(\mathbf{E})}(X) = 0$ , while [L92] would express this as  $\mu_p(X) = 0$ .

**Theorem 4**  $\mu_{\Gamma(\mathcal{C})}(\mathcal{C}) \neq 0$ .

**Proof.** (of Theorem 2). Given a language  $L \in \mathcal{C}$ , we will construct a null cover  $d_k(w)$  of the singleton set  $\{L\}$  as follows: Decide membership of each word  $x$  such that  $\text{pos}(x)$  is a power of 2 and is at most  $|w|$ . Suppose there are  $s$  of these. Output  $2^{s-k}$  if bits  $1, 2, 4, \dots, 2^s$  of  $w$  are all consistent with  $L$ , and output 0 otherwise.

It is easy to check that this yields a cover; we show here that  $d \in (\mathcal{C})$ . Since  $L \in \mathcal{C}$  by hypothesis, we can check membership of any single word  $x$  in time  $f(|x|)$  for some  $f \in \mathcal{C}$ . (This involves no queries to the input  $w$  of  $d$ ). We can in polylog time check all log-many words whose lexicographic position is a power of 2. Also, we query the input at the bits whose positions are powers of 2, generating log-many points in the dependency set. Note that the transitive closure of this set of bits is trivial; no more points are added. ■

**Proof.** (of Theorem 3). Let  $d_{j,k}$  be a 2-DS witnessing that  $X$  is a  $\mathcal{C}$ -union of  $(\mathcal{C})$ -null sets. Let

$$d'_k(w) = \sum_{j=0}^{\infty} d_{j,k+2^{j+1}}(w).$$

We will show  $d'$  is a  $(\mathcal{C})$ -null cover of  $X$ .

It is immediate that  $d'$  is a 1-DS, and that  $d'_k(\lambda) \leq 2^{-k}$ . Now, to see that  $d'$  covers  $X$ , note that if  $\omega$  is any sequence in  $X$ , then  $\omega$  is in some  $X_j$ , and for all  $k$  there is a  $w_k \sqsubseteq \omega$  and  $d_{j,k}(w_k) \geq 1$ . Now note that for all  $k$ ,  $d'_k(w_{k+2^{j+1}}) \geq 1$ , and thus  $d'$  is a null cover of  $X$ . It remains only to show that  $d'$  has  $(\mathcal{C})$ -computations.

Define the computation

$$\hat{d}'_{k,r}(w) = \sum_{j=0}^{\log(r+|w|)} \hat{d}_{j,k+2^{j+1}, r+2^{j+2}}(w).$$

If  $\hat{d}$  is computable in time  $f(\log n)$  with dependency set size also bounded by  $f(\log n)$ , then  $\hat{d}'$  is computable in time  $O(\log n(f(\log n))^2)$ , and is in  $(\mathcal{C})$  by the closure properties assumed of  $\mathcal{F}$ .

Modifying [L92], one can let

$$\sigma = \sum_{j=0}^{\log(r+|w|)} d_{j,k+2^{j+1}}(w),$$

and show  $|d'_k(w) \Leftrightarrow \sigma|$  and  $|\sigma \Leftrightarrow \hat{d}'_{k,r}(w)|$  are each at most  $2^{-(r+1)}$ . ■

**Proof.** (of Theorem 4). We show that if  $d$  is any  $(\mathcal{C})$ -computable density function with  $d(\lambda) < 1$  then we can construct a language in  $\mathcal{C}$  not covered by  $d$ .

As in [L92], find  $q, l$  such that

$$d(\lambda) < q < q + 2^{-l} \leq 1.$$

Let  $a(x) \geq |x| + l + 3$ , and define the desired language  $L$  by

$$\delta(w) = \begin{cases} 1 & \text{if } \hat{d}_{a(w)}(w0) > \hat{d}_{a(w)}(w) + 2^{1-a(w)} \\ 0 & \text{otherwise.} \end{cases}$$

Note that  $\delta$  is clearly a constructor in  $(\mathcal{C})$ , and thus defines a language  $L$  in  $\mathcal{C}$ .

Finally, we have to show  $L$  is not covered by  $d$ , i.e., for all  $n$  we have  $d(\delta^n(\lambda)) < 1$ . Here the inductive argument of [L92] works unchanged. ■

Note that one can also define a measure analogous to our measure on time-bounded classes, using *space* bounds, as opposed to time bounds. In this way, one can obtain a measure  $\mu_{\Gamma(\text{PSPACE})}$  for PSPACE. Mayordomo has previously defined a notion of measure for PSPACE, where, rather than limiting the size of the dependency sets for the machines computing the density functions, instead she requires that the density functions be computed by polylogspace-bounded machines with one-way access to the input [M]. Let us denote her measure  $\mu_{\Phi(\text{PSPACE})}$ .

## 2.4 Elementary Facts Concerning the Measure

We have established that our notion of measure satisfies the measure axioms, but the reader may wonder if enough sets are measurable for the measure to be useful in any setting. Indeed, we show

in this subsection that the class of P-measurable sets is an *extremely* limited class. However, the following theorem and its corollaries (and also the results of the next section) show that measure-zero results can be established for some important classes of sets.

**Theorem 5** *Let  $\mathcal{C} = \text{DTIME}(\mathcal{F})$ . If  $f \in \mathcal{F}$ , then  $\mu_{\Gamma(\mathcal{C})}(\text{DTIME}(f)) = 0$ .*

**Proof.** It will suffice to express  $\text{DTIME}(f(n))$  as a  $\mathcal{C}$ -union of  $(\mathcal{C})$ -null sets. But this is easy. Let  $M_1, M_2, \dots$  be a clocked enumeration of  $\text{DTIME}(f(n))$  machines, and let

$$d_{j,k}(w) = \begin{cases} 0 & \text{if any bits of } w \text{ in positions } \{1, 2, 4, \dots, 2^k\} \text{ is inconsistent with } L(M_j) \\ 2^{\lfloor \log |w| \rfloor - k} & \text{otherwise.} \end{cases}$$

The result follows. ■

**Corollary 6** *For all  $k$ ,  $\mu_{\Gamma(\text{P})}(\text{DTIME}(n^k)) = 0$ .*

**Corollary 7** *Let  $0 < \eta < \epsilon$ , and let  $E_\epsilon$  denote  $\bigcup_{\delta < \epsilon} \text{DTIME}(2^{n^\delta})$ . Then  $\mu_{\Gamma(E_\epsilon)}(E_\eta) = 0$ .*

It is also important for our applications to note that the ‘‘Borel-Cantelli-Lutz Lemma’’ (see [L92]) holds also for our measure:

**Lemma 8** *Let  $\mathcal{C} = \text{DTIME}(\mathcal{F})$ , where  $\mathcal{F}$  contains no superexponential function. Let  $\{X_j\}$  be a collection of (not-necessarily null) sets, where  $X_j$  is covered by  $d_j$  for  $d \in (\mathcal{C})$ . Suppose  $m$  (a ‘‘modulus of convergence’’) is an increasing function of the form  $f(\log n)$  for some  $f \in \mathcal{F}$ , and for all  $k$  we have*

$$\sum_{j=m(k)}^{\infty} d_j(\lambda) < 2^{-k}.$$

Then

$$\mu_{\Gamma(\mathcal{C})}\left(\bigcap_{t=0}^{\infty} \bigcup_{j=t}^{\infty} X_j\right) = 0.$$

■

The proof of this is the same as in [L92], but it is important here to note that for subexponential complexity classes it seems unavoidable that the modulus be *sublinear*. Thus, for the Borel-Cantelli-Lutz Lemma to be applied to subexponential-time classes, the sets  $X_j$  must be covered by a “rapidly-vanishing” collection of covers. This restriction that the values  $d_j(\lambda)$  tend so rapidly toward zero makes it far from obvious that any such functions in  $\mathcal{C}$  can exist; a concrete example is presented in the proof of Theorem 11. The restriction that  $\mathcal{F}$  contain no superexponential functions is necessary; the claim is in fact false for  $\mathcal{F} = 2^{2^{O(n)}}$ . (The proof in [L92] requires the resource-bounding functions to be closed under composition.) It is possible to formulate conditions on  $m$  and  $d$  that allow the lemma to apply to more general classes of resource bounds.

Due to the extremely limited computation power of sublinear-time machines, it is often quite easy to prove that certain sets are *not*, (P)-null. Below, we present one concrete example.

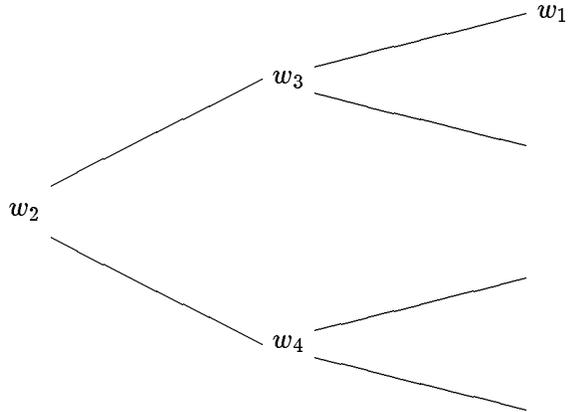
**Theorem 9** *The set SPARSE, of languages having at most a polynomial number of words of any given length, is not, (P)-null in P.*

**Proof.** Let  $d$  be a, (P) density function with  $d(\lambda) < 1$ . We will construct a sparse set in P that is not covered by  $d$ . For clarity of exposition, assume here that  $d = \hat{d}_r$ ; the general case is not significantly more complicated. (Also, it is shown in [AS2] that any, (P) null cover can be replaced by one satisfying this condition; this is discussed more in Subsection 2.5.)

First we show that for any word  $w$ , if  $i$  is the maximum element of  $G_{d,|w|}$ , then  $d(w[0..i]) = d(w[0..j])$  for any  $j$  in the range  $i \leq j \leq |w|$ .

Fix  $w = w_1$ , and let  $w_2 = w[0..i]$  (see Figure 1). Let  $T$  be the complete tree at  $w_2$  of height  $|w_1| \Leftrightarrow i$ ; we will show that  $d$  is constant on  $T$ . By definition of dependency set,  $d$  is constant on the leaves of  $T$ , and by the average law at  $w_2$  this common value equals  $d(w_2)$ . For each  $w_3 \in T$ , we have  $d(w_3)$  is at least the average of  $d$  over the leaves succeeding  $w_3$ , i.e.,  $d(w_3) \geq d(w_2)$ . If  $d(w_3) > d(w_2)$  then by

Figure 1: Tree in Theorem 9



the average law at  $w_2$  there's  $w_4$  at the same level as  $w_3$  such that  $d(w_4) < d(w_2)$ , a contradiction.

Since  $G_{d,i} \subseteq G_{d,|w|}$ , it is now easy to see that for all  $j \notin G_{d,|w|}$ ,  $d(w[0..j \Leftrightarrow 1]) = d(w[0..j])$ .

Since the language  $L \in P$  constructed in the proof of Theorem 4 excludes  $x$  on the all-but-polylog-many places where  $d(L[0..pos(x) \Leftrightarrow 1]0) = d(L[0..pos(x) \Leftrightarrow 1]1)$  and therefore have common value  $d(L[0..pos(x) \Leftrightarrow 1])$ , we conclude  $L$  is sparse. ■

Note this says more than “no, (P) density-system covers SPARSE;” since we constructed an uncovered language in P, we conclude that no, (P) density-system covers the smaller set  $\text{SPARSE} \cap P$ , i.e., that SPARSE has nonzero measure in P. A closer look at this argument also shows that the sparse set that is constructed is, in fact, P-printable ([HY],[AR]), and hence also the class  $K[\log, \text{poly}]$  of languages containing only strings of low time-bounded Kolmogorov complexity [BB, HH] does not have measure zero in P. In contrast, a direct argument shows that for all  $k$ , the set of  $n^k$ -sparse languages (including the set of all tally languages) is, (P)-null.

Observe also that the proof of Theorem 9 works equally well for space-bounded machines, and thus SPARSE is not a measure zero subset of PSPACE using the, (PSPACE) measure. On the

other hand, SPARSE is easily seen to be null using the  $\mu_{\Phi(\text{PSPACE})}$  measure of [M]. The subject of the relation between our PSPACE measure and that of [M] is taken up again in Section 2.5.

Theorem 9 also has the following corollary.

**Theorem 10** *P-uniform  $\text{AC}^0$  is not  $\text{P}$ -measurable.*

**Proof.** Since the class of  $\text{P}$ -printable sets is contained in  $\text{P}$ -uniform  $\text{AC}^0$ , we conclude that  $\text{P}$ -uniform  $\text{AC}^0$  (and therefore non-uniform  $\text{AC}^0$ ) is not  $\text{P}$ -null. It remains only to show that  $\text{P}$ -uniform  $\text{AC}^0$  does not have measure one in  $\text{P}$ . Consider any  $\text{P}$ -density system covering the PARITY language. An argument similar to that of Theorem 9 shows that there is a set  $L$  in  $\text{P}$  that differs from PARITY only on a sparse set, such that  $L$  is not covered. No set that differs from PARITY only on a sparse set can be in  $\text{AC}^0$ , and thus this shows that no  $\text{P}$ -density system can cover  $\text{P} \setminus \text{AC}^0$ . ■

It follows from this that  $\text{P}$ -density functions satisfy the measure axioms even on  $\text{P}$ -uniform  $\text{AC}^0$ , and hence notions of measure can be defined even on intuitively small subsets of  $\text{P}$ . (Recently and independently, Regan and Sivakumar have given a very different argument showing that non-uniform  $\text{AC}^0$  does not have measure zero in  $\text{P}$  [RS].)

We do not view this limitation of our measure as a drawback. Indeed, we conjecture that even  $\text{NTIME}(\log n)$  (a proper subset of  $\text{Dlogtime-uniform AC}^0$ ) is not a measure-zero subset of  $\text{P}$ . Backing up our conjecture is the fact that, although  $\text{NTIME}(\log^{O(1)} n)$  is properly contained in  $\text{DTIME}(2^{\log^{O(1)} n})$ , it is easy to show that if  $\text{NTIME}(\log^{O(1)} n)$  has measure zero in  $\text{DTIME}(2^{\log^{O(1)} n})$ , then  $\text{NP}$  has measure zero in  $\text{DTIME}(2^{n^{O(1)}})$ , which seems to be an unlikely consequence [KM, L93a]. Of course, proving that  $\text{NP}$  does *not* have measure zero in  $\text{DTIME}(2^{n^{O(1)}})$  entails proving that  $\text{P} \neq \text{NP}$ . However, we anticipate that due to the severely limited computational power of  $\text{P}$ -machines, it should be possible to show that some other interesting classes of sets are not  $\text{P}$ -null.

## 2.5 Robustness, Alternative Formulations and Auxiliary Axioms

There are many choices that must be made in making the notion of a measure precise. The definitions in the preceding subsections reflect one set of choices, but it is instructive to consider other ways a definition could have been formulated, to see if the class of measure-zero sets varies under these changes.

Juedes, Lutz and Mayordomo have previously shown that their notion of resource-bounded measure is robust in the face of many modifications of the definition of covers. As a practical matter, when trying to show that a class does *not* have measure zero in  $\text{E}$  or some larger complexity class, it is very useful to know that, in that setting, a null cover can be assumed without loss of generality to satisfy all of the following “niceness” conditions [JLM][JL2][M]:

- A density system  $d_k$  is *exactly computable* if  $d_k = \hat{d}_{k,r}$ .
- A density function is *conservative* if it satisfies the following “conservation” property:  $d(w) = \frac{d(w0) + d(w1)}{2}$ . Although our Definition 1 in this paper<sup>4</sup> requires density functions to be conservative, other papers, such as [L92], for example, require density functions to satisfy *only* the weaker condition  $d(w) \geq \frac{d(w0) + d(w1)}{2}$ .
- If the density system  $d_k$  is of the form  $d_k = 2^{-k}d$  for some density function  $d$ , then we say that  $d_k$  is derived from the *martingale*  $d$ . (Many authors require a martingale to be conservative; we will consider both conservative and nonconservative martingales.) Note that the condition that  $d_k$  be a null cover of  $\omega$  is equivalent to saying that there is no finite upper bound on  $\{d(w) : w \sqsubseteq \omega\}$ , so the lim sup of this sequence is infinite.

<sup>4</sup>An earlier version of this paper did not require the density functions to be conservative; the subtle role that this condition plays in the proof of Theorem 9 did not become clear until later.

- A set  $A$  is *covered in the limit* if there is a martingale  $d$  such that, for all  $\omega \in A$ , the sequence  $\langle d(\omega[0..n]) \rangle$  has a limit of infinity, instead of merely an infinite lim sup.
- A density system is *regular* if  $d_k(z) = 1$  and  $z \sqsubseteq w$  imply  $d_k(w) = 1$ .

When considering measure on subexponential complexity classes, there are additional choices involved in the definition, concerning how (or if) the length of the input is provided, questions concerning how dependency sets should be defined, etc. Several of the proofs of [JLM, JL2, M] showing that their notion of measure is robust under these changes do *not* translate directly to our setting on small measure, which raises the spectre that each of the  $2^5$  combinations of the niceness conditions listed above (not counting additional choices concerning providing the input length, etc.) would give rise to a different notion of measure.

We show in [AS2] that, in fact, only two notions of measure on  $P$  can be defined by varying these parameters. It turns out that any null set can be covered by an exactly-computable martingale, but surprisingly, assuming *any* of the other niceness conditions is equivalent to assuming *all* of them.

That is, it is shown in [AS2] that the notion of measure defined in this paper is equivalent to the definition that results from having the measure-zero sets be covered in the limit by exactly-computable conservative martingales, where the machines that compute the martingales are even more limited than the machines that are considered in this paper. On the other hand, SPARSE is covered (in the lim sup sense) by a non-conservative martingale, so two distinct notions of measure do result.

It is also shown in [AS2] that the plogon measure  $\Phi(\text{PSPACE})$  of [M] is strictly richer than the conservative version of our space measure, but  $\Phi(\text{PSPACE})$  is incomparable with the nonconservative version of our space measure.

The set studied in Section 3 is shown to be null in the most restrictive sense of measure, so

the results of that section hold for all the other measures mentioned here.

### 3 Hard Sets for BPP

It was shown in [BG] that for almost every  $A$ ,  $\text{BPP}^A = P^A$ . In [L93] it was shown that almost every set in  $\text{ESPACE}$  has this property, and thus in particular almost every such set is hard for BPP. Note, on the other hand, that only a measure zero set of languages is hard for  $\Sigma_2^P$  (assuming  $\text{BPP} \neq \Sigma_2^P$ ), and thus the “reason” that a PSPACE-complete set is hard for BPP is in a fundamental way quite different from the “reason” that a random set is hard for BPP.

In this section, we improve the results of [L93] by showing that almost every set in PSPACE and in  $E_\epsilon$  for  $\epsilon > 0$  is hard for BPP, because almost every such set “looks random enough”. As in [L93], we make use of the pseudorandom generators of [NW], although our construction differs from that of [L93] in several fundamental ways.

**Theorem 11** *For almost every  $A \in E_\epsilon$  we have  $\text{BPP} \subseteq P^A$ .*

**Proof.** Define  $H_\eta$  to be the set of languages  $L$  of hardness  $2^{\eta n}$ . That is, for sufficiently large  $n$  any circuit of size  $2^{\eta n}$  errs on at least  $2^{n-1}(1 \leftrightarrow 2^{-\eta n})$  of the words of length  $n$ . For a fixed  $n$ , we let  $H_\eta(n)$  denote the set of languages satisfying this condition at least at the given  $n$ .

Fix positive  $\epsilon$ . We show that for almost every  $A \in E_\epsilon$  and every  $\eta < \min\{\epsilon, 1/3\}$  we have  $E^A \cap H_\eta \neq \emptyset$ . It follows from Lemma 12 that  $\text{BPP} \subseteq P^A$  for any such  $A$ , and hence for almost all  $A \in E_\epsilon$ .

**Lemma 12** *If  $E^A \cap H_\eta \neq \emptyset$ , then  $\text{BPP} \subseteq P^A$ .*

**Proof.** This is a “partially” relativized version of [NW, Theorem 3]. ♣

Let  $0 < \eta < \min\{\epsilon, 1/3\}$ , and let  $b > 1/\epsilon$ . Let  $F(A)$  denote the language

$$\{u : 0^{2^{b|u|}} u \in A\}.$$

Then  $F(A) \in E^A$ . We will show that for almost all  $A \in E_\epsilon$ ,  $F(A)$  has hardness  $2^{\eta n}$ .

Fix a circuit  $\gamma$  with  $n$  inputs. For a word  $u$  of length  $n$ , the set  $\{A : u \in F(A) \text{ iff } \gamma(u) = 1\}$  has (Lebesgue) measure  $1/2$ . That is, if we choose  $A$  at random, the circuit  $\gamma(u)$  is too small to query  $0^{2^{bn}}u \in A$ , so with probability  $1/2$  we have  $(u \in F(A))$  iff  $(\gamma \text{ accepts } u)$ . The events in  $\{(u \in F(A)) \text{ iff } (\gamma \text{ accepts } u)\}_{|u|=n}$  are mutually independent, so we can apply the Chernoff bound to get

$$\left\{ A : \begin{array}{l} \gamma(u) \text{ computes } F(A) \text{ on at} \\ \text{least } 2^{n-1}(1 + 2^{-\eta n}) \text{ words} \\ u \end{array} \right\}$$

has measure less than

$$e^{-(2^{n-\eta n})^2/2^n} \leq 2^{-2^{n/3}} \quad (1)$$

for all sufficiently large  $n$ . Finally, considering all circuits of size  $2^{\eta n}$  and having  $n$  inputs, the set

$$\begin{aligned} X_n &= \left\{ A : \begin{array}{l} \text{some } \gamma(u) \text{ computes} \\ F(A) \text{ on at least} \\ 2^{n-1}(1 + 2^{-\eta n}) \text{ words} \\ u \end{array} \right\} \\ &= \{ A : F(A)_{=n} \notin H_\eta(n) \} \end{aligned}$$

has measure at most  $2^{2^{\eta n}}$  times (1), which is less than  $2^{-2^{n/4}}$  for all large  $n$ . Let  $\text{Ch}_n$  denote the value  $2^{-2^{n/4}}$ .

The set  $X = \limsup X_n$  contains the  $A$  for which  $F(A) \notin H_\eta(n)$  for infinitely many  $n$ , and we wish to show  $X$  is null.

We define a density system  $d_n(w)$  that for all large  $n$  is an upper bound for the conditional probability that a random  $A$  satisfies  $F(A) \notin H_\eta(n)$  given  $A \in C_w$ . For short  $w$ , i.e.,  $|w| < \text{pos}(0^{2^{bn}}0^n) = p$ , define  $d_n(w)$  to be  $\text{Ch}_n$ . If  $|w| \geq p$ , then we will exhaustively generate all extensions of  $w$  corresponding to elements of  $F(A)$  of length  $n$  (i.e., all extensions up to  $\text{pos}(0^{2^{bn}}1^n)$ ), and for each such extension simulate all circuits of size  $2^{\eta n}$  on all strings (inputs) of length  $n$ , and count the number of extensions that can be approximated in that way, to compute the conditional probability. That is, we compute the exact

value of  $\Pr(F(A) \notin H_\eta(n) | C_w)$ . If we were to set  $d_n(w)$  to this value, it would define a non-conservative density system; thus it remains only for us to patch this so that the conservation axiom is satisfied, at the one place where it may fail: the ‘‘seam’’ between the crude Chernoff bound for short  $w$  and the precise conditional probability for large  $w$ . Let  $\alpha$  be defined as:

$$\alpha = \text{Ch}_n \Leftrightarrow \Pr(F(A) \notin H_\eta(n) | C_{w[0..p-1]}).$$

Note that  $\alpha$  depends on  $n$  but not on  $w$ . For  $|w| \geq p$ , define  $d_n(w)$  to be

$$d_n(w) = \alpha + \Pr(F(A) \notin H_\eta(n) | C_w)$$

Observe that if  $|w| > \text{pos}(0^{2^{bn}}1^n)$ , then the conditional probability is either 0 or 1, and it follows easily that  $d_n$  covers  $X_n$ .

It remains to show that  $d_n(w)$  has computations, which we do in pieces. First consider  $\Pr(F(A) \notin H_\eta(n) | C_w)$ , for  $|w| \geq p$ . There are at most  $2^{2^n} = 2^{\log^{1/b}|w|}$  extensions and  $2^{2^{\eta n}}$  circuits to consider, and each simulation takes time  $2^{\eta n}$ . Thus the total time to do all simulations is less than  $2^{\log^\epsilon |w|}$ . The dependency set  $G_{d,|w|,n}$  consists of the polylog( $|w|$ )-many positions of all words of the form  $0^{2^{b|v|}}v$ , so the dependency bound is met also.

Next we consider  $\text{Ch}_n$ . It is not the case that  $\text{Ch}_n$  is exactly computable, since  $\text{Ch}_n = 2^{-2^{n/4}}$  requires  $n/4 + O(1)$  bits to write down as a ‘‘formal sum of powers of two.’’ However, we can write

$$\widehat{\text{Ch}}_{n,r}(w) = \begin{cases} \text{Ch}_n & \text{if } r \geq 2^{n/4}; \\ 0 & \text{otherwise.} \end{cases}$$

If  $r \geq 2^{n/4}$  then expressing  $2^{-2^{n/4}}$  in the desired format requires only writing the exponent  $\Leftrightarrow 2^{n/4}$ , and hence requires at most  $\log r$  bits to write down, and if  $r < 2^{n/4}$  then

$$|\text{Ch}_n \Leftrightarrow \widehat{\text{Ch}}_{n,r}(w)| = \text{Ch}_n = 2^{-2^{n/4}} \leq 2^{-r}.$$

Then set

$$\hat{d}_{n,r}(w) = \begin{cases} \widehat{\text{Ch}}_{n,r}(w) & \text{if } |w| < p, \\ d_n(w) & \text{otherwise.} \end{cases}$$

Finally, we apply the Borel-Cantelli-Lutz Lemma to the density system  $d_n$ , using modulus  $m(n) = 1 + 4 \log n$ . That is, putting  $j = 2^{n/4}$ ,

$$\sum_{n=m(k)} d_n(\lambda) = \sum_{n > 4 \log k} 2^{-2^{n/4}} \leq \sum_{j > k} 2^{-j} \leq 2^{-k},$$

so we conclude  $\mu_{\Gamma(E_\epsilon)}(\limsup X_n) = 0$ . ■

Note that any improvement to a time class smaller than all  $E_\epsilon$ 's would involve showing that each language in BPP has subexponential time complexity.

A similar theorem holds for space bounds:

**Theorem 13** *For almost every  $A \in \text{PSPACE}$  we have  $\text{BPP} \subseteq \text{P}^A$ .*

**Proof.** The proof is similar to the proof in the time-bounded case. We note that a machine for  $d$  only needs bits in positions  $\text{pos}(0^{2^n} 0^n)$  through  $\text{pos}(0^{2^n} 1^n)$ , and a space-bounded machine can store all of these bits. ■

The following corollary of Theorems 11 and 13 was pointed out to us by Jack Lutz:

**Corollary 14** *Let  $\mathcal{C}$  be any of the classes  $E$ ,  $\text{EXP}$ ,  $\text{PSPACE}$  or  $E_\epsilon$ . If  $\mu_{\Gamma(\mathcal{C})}(\text{NP} | \mathcal{C}) \neq 0$  then  $\text{BPP} \subseteq \text{P}^{\text{NP}}$ .* ■

We also show that almost every set  $A$  in  $E$  satisfies  $\text{BPP}^A = \text{P}^A$ , improving the result of [L93] from  $\text{ESPACE}$  to  $E$ .

**Theorem 15** *For almost every  $A \in E$ ,  $\text{BPP}^A = \text{P}^A$ .*

**Proof.** The proof of this theorem is essentially the same as the proofs of Theorems 13 and 11; the only modification is to replace  $H_\eta$  in those proofs with  $H_\eta^A$ : the set of languages  $L$  of “relativized” hardness  $2^{\eta n}$ , using oracle  $A$ . That is, for any  $A \in H_\eta^A$ , for sufficiently large  $n$  any circuit of size  $2^{\eta n}$  agrees with  $F(A)$  on at most  $2^{n-1}(1 + 2^{-\eta n})$  of the words of length  $n$ , where the circuits are now allowed to have “oracle” gates that query the oracle  $A$ . ■

We do not know if the condition of Theorem 15 holds also for  $\text{PSPACE}$  or any  $E_\epsilon$ .

## 4 Pseudorandom Sources

In [L90], Lutz proposed a notion of *source* for BPP. He gave a criterion for a particular sequence to be useful as a substitute for the sequence of independent unbiased coin flips used by a BPP machine. Based on this work, we formulate three intuitive properties a notion of source should have:

**Universality** A single source should “work” for all BPP languages.

**Abundance** The set of sources should have measure 1 at some level of resource-boundedness. This implies that a random sequence of coin flips is a source with probability 1.

**Hardness** If the bits of a source can be obtained in polynomial time, then  $\text{P} = \text{BPP}$ .

The definition in [L90] captures the first two properties, but lacks the third, as one can construct sources in  $\text{AC}^0$  without showing  $\text{P} = \text{BPP}$  [S].

We seek an alternate criterion for a particular computable sequence to be “random enough.” We will capture universality, abundance, and hardness.

Intuitively, a sequence  $A$  is a pseudorandom sequence if it is possible to use  $A$  in place of a sequence of random bits to recognize each BPP language (where it is crucial to the definition that this simulation would work if  $A$  were replaced with a random sequence). Formally, we will say that a sequence  $A = \langle a_i \rangle$  is a *pseudorandom sequence for BPP* if for each  $L \in \text{BPP}$ , there is a bounded-error probabilistic polynomial-time machine accepting  $L$  such that, for each  $x$ ,  $M(x, A) = 1 \Leftrightarrow x \in L$ , where “ $M(x, A)$ ” denotes the result of running machine  $M$  on input  $x$  along the path given by taking the result of the  $i^{\text{th}}$  coin flip to be  $a_i$ . That is, on each input  $x$ , the first polynomially-many bits of the sequence  $A$  are used in place of probabilistic bits.

Stated another way,  $A$  is a pseudorandom sequence for BPP if the set  $T(A) = \{0^i | a_i = 1\}$  is hard for BPP under  $\leq_T^p$  reductions, where the machine  $M$  that reduces BPP language  $L$  to  $T(A)$

also accepts  $L$  when  $M$  is viewed as a BPP machine (i.e., where queries to the oracle are answered randomly). That is,  $A$  is not only hard for BPP, but it is hard because it “looks random.” It follows from [BG] that almost all sequences are pseudorandom sequences for BPP.

**Theorem 16** *Almost every  $A \in \text{ESPACE}$  is a pseudorandom sequence for BPP.*

**Proof.** Theorem 13 shows that for most  $A$  in PSPACE, the language  $F(A) = \{u : 0^{2^{|u|}} u \in A\}$  has hardness  $2^{nm}$ . An essentially equivalent way of restating this is to say that for almost all  $A$  in PSPACE, for almost all  $m$ , the function  $f : \Sigma^m \rightarrow \{0, 1\}$  defined by  $f(u) = \chi_A(0^{m^2+u})$  has hardness  $2^{nm}$ . Thus, informally, almost all tally sets in PSPACE look “random enough” to serve as sources for BPP. Since tally sets in PSPACE are essentially the same thing as sets in ESPACE, a direct argument patterned after the proof of Theorem 13 can be used to show that almost all sets in ESPACE are pseudorandom sequences for BPP. ■

Theorem 16 is in some sense analogous to the main result of [L90], but note that in contrast to [L90], we can make a limited claim of optimality, in the following sense. Theorem 16 shows that there are sources in  $\text{DTIME}(2^{2^{O(n)}})$ . If there is any source  $A$  in  $\text{DTIME}(F(n))$  for some  $F(n) \in 2^{2^{o(n)}}$ , then  $T(A) \in \text{DTIME}(2^{n^{o(1)}})$ , and hence simulating any  $n^k$ -time bounded BPP machine would require at most time  $n^k 2^{(n^k)^{o(1)}}$ , and thus every language in BPP would have time complexity  $2^{n^{o(1)}}$ .

## 5 Conclusions

### 5.1 Does this Relativize?

What is “relativized one-way polylogarithmic space”? One’s initial response is likely to be that this is a ludicrous notion, and that attaching oracles to such limited automata would probably not give rise to a very meaningful class. If followed

further, this line of reasoning suggests that non-relativizing results might be obtained by studying  $\mu_{\Phi(\text{PSPACE})}$ -measure, and similar observations apply to the similarly-limited  $\mu_{\Gamma(\text{P})}$ -measure. Might one be able to show that most sets in P are hard for BPP? Such a result would, of course, require nonrelativizing proof techniques, but if the notion of measure on P does not relativize in a meaningful way, then perhaps it could be used to obtain results of this sort.

Unfortunately, it turns out that these notions of measure *do* relativize in a natural way. When one is dealing with extremely weak models of computation, the question of how to provide access to the oracle is often rather controversial. (A survey of papers discussing the issues involved may be found in [A90].) It is easy to see that, for instance, if one were to use the so-called “Ruzzo-Simon-Tompa” relativization method [RST], then there are oracles  $A$  relative to which  $\text{P}^A$  would have measure zero in  $\text{P}^A$ , and thus this does not constitute a meaningful notion of measure on  $\text{P}^A$ . On the other hand, if one adopts the convention<sup>5</sup> that a  $\text{DTIME}(\log^{O(1)} n)$  machine is permitted to write only queries of length  $\log^{O(1)} n$  on its query tape, then it is straightforward to show that one obtains a measure on  $\text{P}^A$ . Similar observations hold for the measures on other subexponential time classes and on PSPACE.

Although this rules out the prospect of obtaining nonrelativizing results via this notion of measure, a compensating factor is that one obtains measures for  $\text{P}^{\text{SAT}}$  and for all of the other  $\Delta_i$  levels of the polynomial hierarchy. One can also define a measure on the  $\Sigma_k^{\text{P}} \cap \Pi_k^{\text{P}}$  classes, by using covers  $d$  having  $\Sigma_k^{\text{L}}$  machines such that some path outputs, and all non-aborting paths output  $d(w)$ . (For related observations concerning exponential-time classes, see also [M]).

<sup>5</sup>In most other settings, allowing only short queries to the oracle does *not* provide a satisfactory notion of relativization, because one cannot reduce the set  $A$  to itself (because one cannot write the input on the query tape).

## 5.2 Summary

Lutz's resource-bounded measure forms the basis for a large and growing body of interesting work. A limitation of Lutz's notion of measure is that it does not apply to P and other important subexponential time classes. We have remedied that situation by providing a notion of measure that does apply to these classes, and we have used this measure to show, among other things, that almost all sets in  $E_\epsilon$  are hard for BPP, substantially improving a result of [L93].

It is worth noting that Lutz's definitions of resource-bounded measure have evolved somewhat over the years. Similarly, we may expect that as experience is gained, alternative formulations of measure on small time classes may arise. However, we have established that interesting results can be obtained with our current notion of measure, and we look forward to further work in this area.

## Acknowledgments

We gratefully acknowledge helpful discussions and correspondence with Lane Hemaspaandra, Elvira Mayordomo, Jack Lutz, David Juedes, and Ken Regan.

## References

- [A90] E. Allender, Oracles vs proof techniques that do not relativize, *Proc. SIGAL International Symposium on Algorithms*, Lecture Notes in Computer Science 450, 1990, pp. 39–52.
- [AR] E. Allender and R. Rubinfeld, P-printable sets, *SIAM J. Comp.* **17** (1988) 1193–1202.
- [AS] E. Allender and M. Strauss, Towards a Measure for P, DIMACS Technical Report 94-14, 1994.
- [AS2] E. Allender and M. Strauss, in preparation.
- [AJ] C. Álvarez and B. Jenner, On Dlogtime and Polylogtime reductions, *Proc. 11th STACS*, Lecture Notes in Computer Science 775, 1994.
- [Ba] J. Balcázar, Self-Reducibility, *Journal of Computer and System Sciences* **41** (1990) 367–388.
- [BB] J. Balcázar and R. Book, Sets with small generalized Kolmogorov complexity, *Acta Informatica* **23** (1986) 679–688.
- [BG] C. Bennett and J. Gill, Relative to a random oracle,  $P(A) \neq NP(A) \neq Co-NP(A)$  with probability 1, *SIAM J. Comput.* **10** (1981) 96–113.
- [Bu] S. R. Buss, The Boolean formula value problem is in ALOGTIME, *Proc. 19th ACM Symposium on Theory of Computing*, 1987, pp. 123–131.
- [HH] J. Hartmanis and L. Hemachandra, On sparse oracles separating feasible complexity classes, *Information Processing Letters* **28** (1988) 291–296.
- [HY] J. Hartmanis and Y. Yesha, Computation times of NP sets of different densities, *Theoretical Computer Science* **34** (1984) 17–32.
- [JL] D. Juedes and J. Lutz, The complexity and distribution of hard problems, *Proc. 34th FOCS Conference*, pp. 177–185, 1993.
- [JL2] D. Juedes and J. Lutz, Weak completeness in E and  $E_2$ , manuscript.
- [JLM] D. Juedes, J. Lutz and E. Mayordomo, personal communication.
- [KM] S. Kautz and P. Miltersen, Relative to a random oracle, NP is not small, *Proc. 9th Structure in Complexity Theory Conference*, pp. 162–174, 1994.

- [L90a] J. Lutz, Category and measure in complexity classes, *SIAM J. Comput* **19** (1990), 1100–1131.
- [L90] J. Lutz, Pseudorandom Sources for BPP. *J. Computer and System Sciences*, **41** (1990), 307-320.
- [L92] J. Lutz, Almost Everywhere High Nonuniform Complexity, *Journal of Computer and System Sciences* **44** (1992), pp. 220-258.
- [L93] J. Lutz, A Pseudorandom Oracle Characterization of BPP, *SIAM J. Comput.*, **22** 1993, 1075-1086.
- [L93a] J. Lutz, The quantitative structure of exponential time, *Proc. 8th Structure in Complexity Theory Conference*, pp. 158–175, 1993.
- [LM] J. Lutz and E. Mayordomo, Measure, stochasticity, and the density of hard languages, *SIAM J. Comput.* **23** (1994) 762–779.
- [M] E. Mayordomo, Contributions to the Study of Resource-Bounded Measure, PhD Thesis, Universitat Politècnica de Catalunya, Barcelona, 1994. See also [M2], in which a preliminary version of the PSPACE measure appears.
- [M2] E. Mayordomo, Measuring in PSPACE, to appear in *Proc. International Meeting of Young Computer Scientists '92*, Topics in Computer Science series, Gordon and Breach.
- [NW] N. Nisan and A. Wigderson, Hardness vs. Randomness, *Proc. 29th Annual IEEE Symp. on Foundations of Computer Science*, (1988), 2-11.
- [RS] K. Regan and D. Sivakumar, On Resource-bounded Measure and Pseudorandom Generators, manuscript.
- [RST] W. Ruzzo, J. Simon, and M. Tompa, Space-bounded hierarchies and probabilistic computation, *J. Comput. and System Sci.* **28** (1984) 216–230.
- [S] M. Strauss, Normal numbers and sources for BPP, Proc. 12<sup>th</sup> STACS conference, (1995), to appear.