

# RESTRICTIVE ACCEPTANCE SUFFICES FOR EQUIVALENCE PROBLEMS

BERND BORCHERT, LANE A. HEMASPAANDRA AND JÖRG ROTHE

## Abstract

One way of suggesting that an NP problem may not be NP-complete is to show that it is in the promise class UP. We propose an analogous new method—weaker in strength of evidence but more broadly applicable—for suggesting that concrete NP problems are not NP-complete. In particular, we introduce the promise class EP, the subclass of NP consisting of those languages accepted by NP machines that, when they accept, always have a number of accepting paths that is a power of two. We show that FewP, bounded ambiguity polynomial time (which contains UP), is contained in EP. The class EP applies as an upper bound to some concrete problems to which previous approaches have never been successful, for example the negation equivalence problem for OBDDs (ordered binary decision diagrams).

## 1. Introduction

NP languages can be defined via machines that reject when they have zero accepting paths, and accept by having a number of accepting paths that belongs to the set  $\{1, 2, 3, \dots\}$ . A number of researchers have sought to refine the class NP by shrinking the path-cardinality set signifying acceptance, while retaining the requirement that rejection be associated with having zero accepting paths. We will call any such class a *restricted counting class*. The most common restricted counting classes in the literature are random polynomial time (denoted R) and ambiguity-bounded classes such as UP and FewP. Ambiguity-bounded classes will be of central interest to us in the present paper.

Valiant's class UP (unambiguous polynomial time) [29], which is known to differ from P exactly if one-way functions exist [16], has the acceptance set  $\{1\}$ , and so is a restricted counting class. Acceptance sets of the forms  $\{1, 2, 3, \dots, n^{\theta(1)}\}$  and  $\{1, 2\}, \{1, 2, 3\}, \dots$  define, respectively, the class FewP [2] and the classes  $UP_{\leq 2}, UP_{\leq 3}, \dots$  [3], and thus these too are restricted counting classes. (Note:  $UP \subseteq UP_{\leq 2} \subseteq UP_{\leq 3} \subseteq \dots \subseteq UP_{\theta(1)} \subseteq \text{FewP} \subseteq \text{NP}$ , where  $UP_{\theta(1)} = \bigcup_{k \geq 1} UP_{\leq k}$ .) These classes are also connected to the existence of one-way functions and have been extensively studied in a wide variety of contexts, such as class containments [14, 23], complete sets [21], boolean hierarchy equivalences [19], complexity-theoretic analogs of Rice's theorem [7, 20], and upward separations [26].

---

This work was supported in part by grants NSF-CCR-9322513, NSF-INT-9513368/DAAD-315-PRO-fo-ab, and NSF-INT-9815095/DAAD-315-PPP-gü-ab, a NATO Postdoctoral Science Fellowship from the Deutscher Akademischer Austauschdienst ("Gemeinsames Hochschulsonderprogramm III von Bund und Ländern"), and a Heisenberg Fellowship from the Deutsche Forschungsgemeinschaft. This work was done in part while the second author was visiting Friedrich-Schiller-Universität Jena, and while the third author was visiting the University of Rochester and Le Moyne College.

Received 10 June 1999, revised 4 November 1999; published 14 March 2000.

2000 Mathematics Subject Classification 68Q15, 68Q10, 03D15

© 2000, Bernd Borchert, Lane A. Hemaspaandra and Jörg Rothe

Of course, the litmus test of NP refinements such as R, UP,  $UP_{\leq k}$ , and FewP is *the extent to which they allow us to refine the upper bounds on the complexity of natural NP problems*. Of these classes, R and UP have been quite successful in this regard. R is well-known to provide an upper bound on the complexity of primality testing. UP is well-known to provide an upper bound on the complexity of (a language version of) the discrete logarithm problem, and UP (indeed  $UP \cap \text{coUP}$ ) is also well-known to provide an upper bound on the complexity of primality testing.

However, there are certain NP problems whose richness of structure has to date defied attempts to put them in UP or even FewP, yet that nonetheless intuitively seem to use less than the full generality of NP's acceptance mechanism. To try to categorize these problems, we introduce the class EP, which is intermediate between FewP and NP:  $\text{FewP} \subseteq \text{EP} \subseteq \text{NP}$ . In particular, EP is the NP subclass whose acceptance set is  $\{2^i \mid i \in \mathbb{N}\}$ ,  $\mathbb{N} = \{0, 1, 2, 3, \dots\}$ .

In Section 2, we provide improved upper bounds on the complexity of the problems OBDD (ordered binary decision diagram) negation equivalence, 2-dag interchange equivalence, and boolean negation equivalence. These three problems are trivially in, respectively, NP, NP, and  $\text{NP}^{\text{NP}}$ . We provide upper bounds of, respectively, EP, EP, and  $\text{EP}^{\text{NP}}$ . The problems are not known to belong to (and do not seem obviously to belong to), respectively, FewP, FewP, and  $\text{FewP}^{\text{NP}}$ .

In Section 3, we prove a general result regarding containment of FewP in certain restricted counting classes. In particular, we obtain a sufficient condition for establishing when restricted counting classes contain FewP. From our result it follows that EP contains FewP; moreover, our result subsumes as special cases some previously known results from the literature. In Section 4, we list open questions related to our work.

## 2. Concrete problems and EP

In this section, we provide concrete problems known to be in NP (or  $\text{NP}^{\text{NP}}$ ), and we prove they are in fact in EP (or  $\text{EP}^{\text{NP}}$ ). We now define the class EP (mnemonic: the number of accepting computation paths is restricted to being either 0 or some power (some exponentiation) of 2). For any nondeterministic polynomial-time Turing machine  $N$  and any string  $x$ , let  $\#\text{acc}_N(x)$  denote the number of accepting computation paths of  $N$  on input  $x$ . Our alphabet  $\Sigma$  will be  $\{0, 1\}$ . For any string  $x \in \Sigma^*$ , let  $|x|$  denote the length of  $x$ .

**Definition 2.1.** EP denotes the class of all languages  $L$  for which there is a nondeterministic polynomial-time Turing machine  $N$  such that, for each input  $x \in \Sigma^*$ ,

$$\begin{aligned} x \notin L &\implies \#\text{acc}_N(x) = 0, \text{ and} \\ x \in L &\implies \#\text{acc}_N(x) \in \{2^i \mid i \in \mathbb{N}\}. \end{aligned}$$

We pass on a comment of an anonymous referee who noted that EP is probably not robust under definitional perturbations; for example, the analog of EP based on any power of 3 is probably a different class. However, in Section 3, we will prove a result general enough so as to apply also to many restricted counting classes other than EP (including the just-mentioned analog of EP).

Now, consider the following well-known problem.

**Problem:** Boolean negation equivalence (BNE) (see [17] and [10]).

**Input:** Two boolean functions (input as boolean formulas using variable names and the symbols  $\{\wedge, \vee, \neg, (, )\}$ ),  $f(x_1, \dots, x_n)$  and  $g(x_1, \dots, x_n)$ , over the same  $n$  boolean variables.

**Question:** Are  $f$  and  $g$  negation equivalent? That is, can one negate some of the inputs of  $g$  such that  $f$  and the modified function  $g'$  are equivalent? (The notion of boolean function equivalence underlying the definition of negation equivalence is the standard one. Two boolean functions (over the same variables) are equivalent if they have the same truth value for every assignment to their variables. Testing the equivalence of pairs of boolean formulas is in coNP.) For concreteness as a language problem,  $\text{BNE} = \{(f, g) \mid f \text{ and } g \text{ are negation equivalent}\}$ .

For example, the two boolean functions described by the formulas  $x_1 \vee x_2 \vee x_3$  and  $x_1 \vee \neg x_2 \vee \neg x_3$  are negation equivalent by negating  $x_2$  and  $x_3$ . Regarding lower bounds, Borchert, Ranjan, and Stephan [10] have shown that the problem USAT [6] polynomial-time many-one reduces to BNE, so BNE is coNP-hard. Regarding upper bounds,  $\text{BNE} \in \text{NP}^{\text{NP}}$  [10] and  $\text{BNE} \in \text{coAM}^{\text{NP}}$  (combining [10] and [1]). It follows from the latter that BNE is not  $\text{NP}^{\text{NP}}$ -complete unless the polynomial hierarchy collapses ([1], in the light of [10, 27]). Interestingly, neither of these two upper bounds ( $\text{NP}^{\text{NP}}$  and  $\text{coAM}^{\text{NP}}$ ) is known to imply the other.

We now prove that  $\text{BNE} \in \text{EP}^{\text{NP}}$ , which is neither known to imply nor known to be implied by the  $\text{coAM}^{\text{NP}}$  upper bound, but which clearly improves the  $\text{NP}^{\text{NP}}$  upper bound as  $\text{EP}^{\text{NP}} \subseteq \text{NP}^{\text{NP}}$ .

**Theorem 2.2.**  $\text{BNE} \in \text{EP}^{\text{NP}}$ .

*Proof.* Suppose that a given instance of BNE consists of  $f$  and  $g$ , each over the variables  $x_1, \dots, x_n$ . A negation of some of the input variables of  $g$  as in the definition of BNE can be represented by a vector  $\vec{v} = (c_1, \dots, c_n)$  in the vector space  $\text{GF}(2)^n$ , where each  $c_i$  is either 0 or 1 and  $c_i = 1$  means that the variable  $x_i$  will be negated. Let  $g_{\vec{v}}$  be the boolean function resulting from  $g$  after the application of the negations described by  $\vec{v}$ ; that is,  $g_{\vec{v}}(\vec{u}) = g(\vec{v} + \vec{u})$ . Now it is easy to see (double negation equals identity, and addition in  $\text{GF}(2)^n$  is associative) that, for each fixed boolean function  $g$ , the set of negation vectors  $\vec{v}$  such that  $g$  equals  $g_{\vec{v}}$  is a linear subspace  $V_g$  of  $\text{GF}(2)^n$ . It is not hard to see that if  $\vec{w}$  is any negation vector such that  $f = g_{\vec{w}}$ , then the affine subspace  $\vec{w} + V_g$  is the set of all negation vectors witnessing the negation equivalence of  $f$  and  $g$ .

Of course,  $\vec{w} + V_g$  will be of the same cardinality as the subspace  $V_g$  (as addition by  $\vec{w}$  induces a bijection between  $\text{GF}(2)^n$  and itself), and as an  $\ell$ -dimensional vector space over the field  $\text{GF}(2)$  has exactly  $2^\ell$  vectors,  $\vec{w} + V_g$  will contain exactly  $2^m$  vectors, where  $m$  is the dimension of  $V_g$ . So the following nondeterministic program shows that BNE is in EP with an NP oracle: read the two input functions  $f$  and  $g$  (checking that they are both over the same number of variables and that the variables have the same naming scheme), guess a negation vector  $\vec{v}$  and accept if and only if the oracle confirms that  $f$  is equal to  $g$  altered by the negation vector  $\vec{v}$ . This shows that BNE is in  $\text{EP}^{\text{NP}}$ , since if  $f$  and  $g$  are not negation equivalent, then there is no accepting path, and otherwise there are exactly  $2^m$  accepting paths, where  $m$  is the dimension of the affine subspace discussed above.  $\square$

There are ways of describing boolean functions such that the equivalence problem is in P. The most prominent such way is by means of ordered binary decision diagrams (OBDDs). So, essentially by the same type of discussion found in the proof of Theorem 2.2, the

following computational problem, OBDD negation equivalence, is in (nonrelativized) EP: given a pair  $(e, f)$  of OBDDs, are the boolean functions described by  $e$  and  $f$  negation equivalent?

(It was Fortune, Hopcroft, and Schmidt [15] who proved that equivalence for OBDDs is in P. OBDDs have recently become a structure of interest to theoretical computer scientists in a variety of settings; see, for example, [28, 13]. For general background on OBDDs see, for example, the survey by Bryant [11].)

**Corollary 2.3.** *OBDD negation equivalence*  $\in$  EP.

Consider the following graph-theoretic problem, which we will call *2-dag interchange equivalence*. First, we need some definitions.

A *2-dag* is a directed acyclic graph having a unique root, and satisfying the condition that every node either has no successor, or has two ordered outgoing edges, where one edge is labeled 0, the other edge is labeled 1, and the two edges may lead to the same successor node. Each node  $v$  of a given 2-dag is assigned a depth, namely the length of a shortest path from the root to  $v$ . (Thus, for example, the root is at depth zero.)

For any 2-dags  $F$  and  $G$ , we say that  $F$  and  $G$  are *isomorphic* if there is a bijective mapping  $\pi$  from the nodes of  $F$  onto the nodes of  $G$  such that for every two nodes  $v, w$  in  $F$  it holds that, for  $i \in \{0, 1\}$ , an edge labeled  $i$  leads from  $v$  to  $w$  in  $F$  if and only if an edge labeled  $i$  leads from  $\pi(v)$  to  $\pi(w)$  in  $G$ .

For any 2-dags  $F$  and  $G$ , we say that  $F$  and  $G$  are *interchange equivalent* if there is a (possibly empty) set of nonnegative integers  $\{d_1, \dots, d_m\}$  such that, for each depth  $d \in \{d_1, \dots, d_m\}$  and for each node  $v$  of depth  $d$  in  $G$ , if the labels of the two outgoing edges of  $v$  (if any such edges exist) are interchanged, then the modified 2-dag  $G'$  is isomorphic to  $F$ . The corresponding computational problem (2-dag interchange equivalence) is: given two 2-dags  $F$  and  $G$ , are  $F$  and  $G$  interchange equivalent?

**Theorem 2.4.** *2-dag interchange equivalence*  $\in$  EP.

*Proof.* This proof is reminiscent of the proof of Theorem 2.2. Let  $F$  and  $G$  be any given 2-dags. If they differ in their maximum depths then they are not interchange equivalent. Otherwise, let  $b$  be the maximum depth of the nodes in  $G$ . Any set of nonnegative integers  $\{d_1, \dots, d_m\}$  dictates a length  $b$  vector (whose positions we will index as  $0, 1, \dots, b-1$ ) via having a “1” at positions  $d_1, \dots, d_m$ , and a “0” at the other positions (numbers  $d_i$  that are greater than or equal to  $b$  do not matter). As in the proof of Theorem 2.2, it is easy to see that the set of length  $b$  vectors that yield a 2-dag  $G'$  isomorphic to  $G$  is a linear subspace of the vector space  $\text{GF}(2)^b$ . Hence, the dimension of this linear subspace is some power of two. Also, the set of all length  $b$  vectors that turn  $G$  into a 2-dag  $G'$  isomorphic to  $F$  is an affine subspace of  $\text{GF}(2)^b$  with the same dimension.

Note that the isomorphism problem for 2-dags can be solved in deterministic polynomial time, the ordering making this job easy.

Summarizing, an EP algorithm for 2-dag interchange equivalence proceeds as follows: given two 2-dags  $F$  and  $G$ , guess a vector  $v$  from  $\text{GF}(2)^b$ ; for each  $v$  guessed, apply  $v$  to  $G$ , which gives  $G'$ ; check whether  $G'$  and  $F$  are isomorphic; accept if this is the case, otherwise reject. □

Note also that the 2-dag interchange equivalence problem can easily be  $\leq_m^P$ -reduced to directed graph isomorphism via a reduction that maps 2-dags that are trees to (directed) trees, and many standard  $\leq_m^P$ -reductions from directed graph isomorphism to graph isomorphism

map directed trees to trees (see [24]). Since tree isomorphism is in P [24], clearly 2-dag interchange equivalence for trees is in P. Though, as just noted, some restricted subclasses of 2-dags have the property that their interchange equivalence problems are in P, the authors know of no P algorithm for the general 2-dag interchange equivalence problem. Even refined group-theoretic or graph-theoretic methods such as those described in [22, 25], for example, do not seem to be applicable.

### 3. Location of EP

We state a general result that our technique gives, regarding the containment of FewP in restricted counting classes. We need some additional definitions.

**Definition 3.1.** Let  $S$  be any set of positive integers. Define the *restricted counting class*  $RC_S$  as follows.  $L \in RC_S$  if and only if there exists a nondeterministic polynomial-time Turing machine  $N$  such that, for every  $x \in \Sigma^*$ ,

1. if  $x \in L$  then  $\#acc_N(x) \in S$ , and
2. if  $x \notin L$  then  $\#acc_N(x) = 0$ .

For example, Valiant’s extensively studied class UP equals  $RC_{\{1\}}$ , and, for each  $k \geq 2$ , the class  $ModZ_kP$  of Beigel, Gill, and Hertrampf [5] equals  $RC_{\mathbb{N}-\{a \mid (\exists b \in \mathbb{N}) [a=b \cdot k]\}}$ . Note that, for every nonempty set  $S$  of positive integers, UP is clearly contained in  $RC_S$ . Theorem 3.4 below will establish a condition on sets  $S$  sufficient to ensure that even FewP is contained in  $RC_S$ .

A set is non-gappy if it has only small holes.

**Definition 3.2.** Let  $S$  be any set of positive integers. We say that  $S$  is *non-gappy* if  $S \neq \emptyset$  and  $(\exists k > 0)(\forall n \in S)(\exists m \in S)[m > n \wedge m/n \leq k]$ .

**Definition 3.3.** [18] Let  $L$  be any subset of  $\Sigma^*$ . We say that  $L$  is *P-printable* if there is a deterministic Turing machine  $M$  that runs in polynomial time such that, for every nonnegative integer  $n$ ,  $M(0^n)$  prints out the set  $\{x \mid x \in L \wedge |x| \leq n\}$ .

**Theorem 3.4.** *Let  $T$  be any set of positive integers such that  $T$  has a non-gappy, P-printable subset. Then  $FewP \subseteq RC_T$ .*

(Though this result is stated in a relatively general format, we mention in passing that even the restriction employed can be relaxed to the case of nonempty sets of positive integers for which, for some uniform constant, given any integer in the set finding another larger but at most multiplicatively-constantly-larger integer in the set is a polynomial-time task. One can even slightly relax the growth rate, but one has to be very careful to avoid a ‘bootstrapping’ growth-explosion effect via clocking growth rates always with respect to the input. In any case, we feel that the current statement of the theorem is general enough to capture the generality of the result without being so technical as to obscure its essence.)

Our proof technique builds (for example, by adding a rate-of-growth argument) on that used by Cai and Hemachandra [12] to prove that  $FewP \subseteq \oplus P$ , where  $\oplus P$  as is standard is the class of languages  $L$  such that, for some nondeterministic polynomial-time Turing machine  $N$ , on each  $x$  it holds that  $x \in L \iff \#acc_N(x) \equiv 1 \pmod{2}$ . We note that Köbler, Schöning, Toda, and Torán [23] interestingly built on that technique in their

proof that  $\text{FewP} \subseteq \text{C}_{\leq}\text{P}$ , where  $\text{C}_{\leq}\text{P}$  [30] is the class of languages  $L$  such that there is a polynomial-time function  $f$  and a nondeterministic polynomial-time Turing machine  $N$  such that for each  $x$ ,  $x \in L$  if and only if  $\#acc_N(x) = f(x)$ . More recently, this proof technique was useful in establishing a  $\text{UP}_{\Theta(1)}$ -Turing-hardness lower bound for nontrivial counting properties of boolean circuits [20], a result that represents the strongest current complexity-theoretic analog of Rice's theorem.

We now give the proof of Theorem 3.4.

*Proof.* Let  $S$  be a non-gappy, P-printable subset of  $T$ . Let  $k > 0$  be, for  $S$ , some constant satisfying Definition 3.2.

Let  $L$  be any language in  $\text{FewP}$ . Let  $\hat{N}$  be a machine witnessing that  $L \in \text{FewP}$ , and let  $p$  be a polynomial bounding the nondeterministic ambiguity of  $\hat{N}$ ; that is, for each input  $x$ ,  $\#acc_{\hat{N}}(x) \leq p(|x|)$ . To show that  $L \in \text{RC}_T$ , we describe a nondeterministic polynomial-time Turing machine  $N$  that accepts  $L$  via the  $\text{RC}_T$  acceptance mechanism.

On input  $x$ , machine  $N$  chooses  $p(|x|)$  natural numbers  $c_1, c_2, \dots, c_{p(|x|)}$  as follows. Initially, we assume that  $c_1$ , which is defined to be the least element of  $S$ , is hard-coded into the program of  $N$ . Successively, for  $i = 2, \dots, p(|x|)$ , machine  $N$  operates on input  $x$  as follows.

- Let  $c_1, \dots, c_{i-1}$  be the constants that have already been chosen. Define  $b_i = \binom{i}{1}c_1 + \binom{i}{2}c_2 + \dots + \binom{i}{i-1}c_{i-1}$ .
- Let  $a_i$  be the least element of  $S$  such that  $b_i \leq a_i$ .
- Set  $c_i = a_i - b_i$ .

After having chosen these constants,  $N$  (still on input  $x$ ) will operate as follows. It will nondeterministically guess an integer  $i \in \{1, 2, \dots, p(|x|)\}$  and, for each  $i$  guessed, nondeterministically guess each (unordered)  $i$ -tuple of distinct paths of  $\hat{N}(x)$ . On each path  $\alpha$  resulting from such a guess series,  $N(x)$  sees whether the  $i$  paths of  $\hat{N}(x)$  that were guessed on  $\alpha$  are all accepting paths. If all are accepting paths, then path  $\alpha$ , via trivial nondeterministic guesses, splits itself into  $c_i$  accepting paths. On the other hand, if at least one of the  $i$  guessed paths is a rejecting path, then path  $\alpha$  simply rejects. This completes the description of  $N$ .

The intuition behind the construction of  $N$  is that for each input  $x$  the following holds.  $N(x)$  has  $c_1$  accepting paths for each accepting path of  $\hat{N}(x)$ ;  $N(x)$  has  $c_2$  additional accepting paths for each pair of distinct accepting paths of  $\hat{N}(x)$ ; and so on. So, if  $x \in L$ ,  $N(x)$  has  $c_{\#acc_{\hat{N}}(x)}$  additional accepting paths for the (one)  $\#acc_{\hat{N}}(x)$ -tuple of distinct accepting paths of  $\hat{N}(x)$ . However, if for some  $z$  with  $\#acc_{\hat{N}}(x) < z \leq p(|x|)$  a  $z$ -tuple of distinct paths of  $\hat{N}(x)$  was guessed on a path  $\alpha$  of  $N(x)$ , then  $\alpha$  must contain a rejecting path of  $\hat{N}(x)$ , and thus  $N(x)$  will have no accepting paths related to  $c_z$ . This intuition is expressed formally thus:

$$\#acc_N(x) = \binom{\#acc_{\hat{N}}(x)}{1}c_1 + \binom{\#acc_{\hat{N}}(x)}{2}c_2 + \dots + \binom{\#acc_{\hat{N}}(x)}{\#acc_{\hat{N}}(x)}c_{\#acc_{\hat{N}}(x)}.$$

Assume that  $x \in L$ . Thus,  $0 < \#acc_{\hat{N}}(x) \leq p(|x|)$ . Since  $c_{\#acc_{\hat{N}}(x)}$  was chosen so that

$$\#acc_{\hat{N}}(x) = 1 \implies \#acc_N(x) = c_1, \text{ and}$$

$$\#acc_{\hat{N}}(x) \geq 2 \implies \#acc_N(x) = b_{\#acc_{\hat{N}}(x)} + c_{\#acc_{\hat{N}}(x)} = a_{\#acc_{\hat{N}}(x)},$$

and since both  $c_1$  and  $a_{\#acc_{\hat{N}}(x)}$  are elements of  $S$ , it follows that  $\#acc_N(x) \in T$ . On the other hand, if  $x \notin L$  then  $\#acc_{\hat{N}}(x) = 0$ , and so  $\#acc_N(x) = 0$ .

So now, to prove that  $L \in \text{RC}_T$ , it suffices to establish an exponential (in  $|x|$ ) upper bound on the value of  $\max_{i \leq p(|x|)} c_i$ .

We will consider, for  $j \geq 2$ , what bounds hold on the value of  $c_j$ . By construction of  $N$  and since  $S$  is non-gappy, we have  $c_j \leq a_j \leq kb_j$ . Regarding the latter inequality, note that  $b_j$  is not necessarily an element of  $S$ . However, for each  $j$ , we have  $c_1 \leq b_j$ ; so for each  $j$ , there exists a  $\hat{b}_j \in S$  such that  $\hat{b}_j \leq b_j$  and  $\hat{b}_j$  is the greatest such integer in  $S$ . Since  $a_j$  is defined to be the least element of  $S$  such that  $b_j \leq a_j$ , we have  $a_j \leq k\hat{b}_j \leq kb_j$ .

From the above and the definition of  $b_j$ , we have:

$$c_j \leq k \left( \binom{j}{1} c_1 + \binom{j}{2} c_2 + \cdots + \binom{j}{j-1} c_{j-1} \right) \leq k(j-1) \binom{j}{\lfloor \frac{j}{2} \rfloor} \max_{1 \leq i \leq j-1} c_i. \quad (1)$$

The factor  $j-1$  in inequality (1) is the number of terms in  $b_j$ , and the coefficient  $\binom{j}{\lfloor \frac{j}{2} \rfloor}$  is the biggest binomial coefficient of any term in  $b_j$ .

Recall that once we were given  $S \subseteq T$  we fixed  $k$ . For all sufficiently large  $j$  the following holds:

$$k(j-1) \binom{j}{\lfloor \frac{j}{2} \rfloor} \leq \left( \binom{j}{\lfloor \frac{j}{2} \rfloor} \right)^2 \leq (2^j)^2. \quad (2)$$

In particular, let  $j_{\text{bad}} = j_{\text{bad}}(k)$  be the largest  $j$  for which the above inequality fails to hold. (If it always holds, set  $j_{\text{bad}} = 1$ .) Let  $I_{\text{bad}} = \max_{1 \leq i \leq j_{\text{bad}}} c_i$ . From inequalities (1) and (2), we clearly have that

- (a) for  $j > j_{\text{bad}}$ ,  $c_j \leq I_{\text{bad}} \cdot \prod_{j_{\text{bad}} < i \leq j} 2^{2i}$ , and
- (b) for  $j \leq j_{\text{bad}}$ ,  $c_j \leq I_{\text{bad}}$ .

This implies that  $c_j = 2^{\mathcal{O}(j^2)}$ . Thus, for the fixed  $k$  associated with  $S \subseteq T$ , the value of  $\max_{i \leq p(|x|)} c_i$  is indeed bounded by an exponential function in  $|x|$ . Hence,  $L \in \text{RC}_T$ , and thus  $\text{FewP} \subseteq \text{RC}_T$ .  $\square$

From Theorem 3.4 it immediately follows that  $\text{FewP} \subseteq \text{EP}$ , since  $\text{EP} = \text{RC}_{\{2^i \mid i \in \mathbb{N}\}}$  and  $\{2^i \mid i \in \mathbb{N}\}$  is clearly a P-printable, non-gappy set.

**Corollary 3.5.**  $\text{FewP} \subseteq \text{EP}$ .

The comments attached to our on-line technical report version [8] give some of the history of the proof of our results, and of some valuable comments made by R. Beigel, in particular that  $\text{FewP}$  is also contained in the  $\text{EP}$  analog based on any integer  $n$  (note that the acceptance sets for such classes are P-printable and non-gappy).

Note that since Corollary 3.5 in fact relativizes, and as it is well-known that there are relativized worlds in which  $\text{UP}$  and  $\text{FewP}$  differ, it follows immediately that there are relativized worlds in which  $\text{EP}$  is not equal to  $\text{UP}$ . We note also that it is immediate from the definition that the class  $\text{EP}$  is closed under intersection.

Cai and Hemachandra's result  $\text{FewP} \subseteq \oplus\text{P}$  [12] has been generalized to  $\text{FewP} \subseteq \text{ModZ}_k\text{P}$ , for each  $k \geq 2$  [5]. This generalization also follows as a special case of Theorem 3.4 since  $\text{ModZ}_k\text{P} = \text{RC}_{\mathbb{N} - \{a \mid (\exists b \in \mathbb{N}) [a = b \cdot k]\}}$  as mentioned above.

Proposition 3.6 below shows that  $\text{EP}$  is contained in  $\text{C}_{\subseteq}\text{P}$ . (After seeing an earlier draft of this paper, R. Beigel communicated to the authors in February, 1998 that he observed that  $\text{EP}$  is even contained in the class  $\text{LWPP}$  [14]. Since it is known from the work of Fenner, Fortnow, and Kurtz [14] that  $\text{SPP} \subseteq \text{LWPP} \subseteq \text{C}_{\subseteq}\text{P}$ , this improves upon our result, and in particular shows that  $\text{EP}$  is  $\text{PP}$ -low (that is,  $\text{PP} = \text{PP}^{\text{EP}}$ ), where  $\text{PP}$  denotes probabilistic

polynomial time.) In the light of Proposition 3.6, Corollary 3.5 improves upon the result of Köbler *et al.* that  $\text{FewP} \subseteq \text{C=P}$  [23]—an improvement that seems neither to imply, nor to be implied by, other improvements of their result such as  $\text{Few} \subseteq \text{SPP}$  ([23]; see also [14]).

**Proposition 3.6.**  $\text{EP} \subseteq \text{C=P}$ .

*Proof.* Let ES (which is the nonpromise version of EP) denote the class of all languages  $L$  for which there is a nondeterministic polynomial-time Turing machine  $N$  such that, for each input  $x \in \Sigma^*$ ,  $x \in L \iff \#\text{acc}_N(x) \in \{2^i \mid i \in \mathbb{N}\}$ . Note that, clearly,  $\text{EP} \subseteq \text{ES}$ . However, note that  $\text{ES} = \text{C=P}$ , as we now argue.  $\text{ES} \subseteq \{L \mid (\exists A \in \text{C=P})[L \leq_d^p A]\}$  is immediately clear from the definitions, where  $\leq_d^p$  is polynomial-time disjunctive reducibility. So  $\text{ES} \subseteq \text{C=P}$ , as it is known that  $\text{C=P} = \{L \mid (\exists A \in \text{C=P})[L \leq_d^p A]\}$  [4]. To show that  $\text{C=P} \subseteq \text{ES}$ , consider a  $\text{C=P}$  machine  $M$ , and the function  $f$  giving the number of paths on which it would accept, and a polynomial  $p$  such that on all inputs of each length  $n$ ,  $M$  runs for at most  $p(n)$  steps. (Our model is that  $M$  makes binary branching moves, so on each input of length  $n$  machine  $M$  has at most  $2^{p(n)}$  accepting paths.) Consider the EP machine that on input  $x$  has  $2^{1+p(|x|)} - f(x)$  paths that immediately accept, and that also has paths that simulate the  $\text{C=P}$  machine. Note that this machine accepts the  $\text{C=P}$  language.  $\square$

For a detailed discussion of the relation of EP to other complexity classes, and for open questions in addition to those presented in Section 4, we refer the reader to the technical report and conference versions of this paper [8, 9].

#### 4. Open questions

Does EP equal NP? It would be nice to give evidence that such an equality would, for example, collapse the polynomial hierarchy. However,  $\text{UP} \subseteq \text{EP} \subseteq \text{NP}$ , and at the present time, it is open whether even the stronger assumption  $\text{UP} = \text{NP}$  implies any startling collapses. Also, does EP, in contrast to most promise classes, have complete sets? We conjecture that EP lacks complete sets (of course, if EP equals NP then EP has complete sets).

EP clearly is closed under conjunctive reductions and under disjoint union, and (thus) under intersection. Is EP closed under disjunctive reductions or union?

*Acknowledgements.* We thank two referees, R. Beigel, D. Kratsch, H. Müller, F. Stephan, and G. Wechsung for interesting comments and discussions.

#### References

1. M. AGRAWAL and T. THIERAUF, ‘The boolean isomorphism problem’, Proceedings of the 37th IEEE Symposium on Foundations of Computer Science (IEEE Computer Society Press, 1996) 422–430. 88, 88
2. E. ALLENDER and R. RUBINSTEIN, ‘P-printable sets’, *SIAM J. Comput.* 17 (1988) 1193–1202. 86
3. R. BEIGEL, ‘On the relativized power of additional accepting paths’, Proceedings of the 4th Structure in Complexity Theory Conference (IEEE Computer Society Press, 1989) 216–224. 86

4. R. BEIGEL, R. CHANG and M. OGIWARA, 'A relationship between difference hierarchies and relativized polynomial hierarchies', *Math. Systems Theory* 26 (1993) 293–310. 93
5. R. BEIGEL, J. GILL and U. HERTRAMPF, 'Counting classes: thresholds, parity, mods, and fewness', Proceedings of the 7th Annual Symposium on Theoretical Aspects of Computer Science, Lecture Notes in Comput. Sci. 415 (ed. C. Choffrut and T. Lengauer, Springer-Verlag, 1990) 49–57. 90, 92
6. A. BLASS and Y. GUREVICH, 'On the unique satisfiability problem', *Information and Control* 55 (1982) 80–88. 88
7. B. BORCHERT and F. STEPHAN, 'Looking for an analogue of Rice's theorem in circuit complexity theory', *Math. Logic Quart.* To appear. 86
8. B. BORCHERT, L. HEMASPAANDRA and J. ROTHE, 'Powers-of-two acceptance suffices for equivalence and bounded ambiguity problems', Tech. Rep. TR96-045, Electronic Colloquium on Computational Complexity, August 1996 <http://www.eccc.uni-trier.de/eccc/>. 92, 93
9. B. BORCHERT, L. HEMASPAANDRA and J. ROTHE, 'Restrictive acceptance suffices for equivalence problems', Proceedings of the 12th Conference on Fundamentals of Computation Theory, Lecture Notes in Comput. Sci. 1684 (ed. G. Ciobanu and G. Paun, Springer Verlag, 1999) 124–135. 93
10. B. BORCHERT, D. RANJAN and F. STEPHAN, 'On the computational complexity of some classical equivalence relations on boolean functions', *Theory Comput. Syst.* 31 (1998) 679–693. 88, 88, 88, 88, 88
11. R. BRYANT, 'Symbolic boolean manipulation with ordered binary decision diagrams', *ACM Computing Surveys* 24 (1992) 293–318. 89
12. J. CAI and L. HEMACHANDRA, 'On the power of parity polynomial time', *Math. Systems Theory* 23 (1990) 95–106. 90, 92
13. J. FEIGENBAUM, S. KANNAN, M. VARDI and M. VISWANATHAN, 'Complexity of problems on graphs represented as OBDDs', Proceedings of the 15th Annual Symposium on Theoretical Aspects of Computer Science, Lecture Notes in Comput. Sci. 1373 (ed. M. Morvan, C. Meinel and D. Krob, Springer-Verlag, 1998) 216–226. 89
14. S. FENNER, L. FORTNOW and S. KURTZ, 'Gap-definable counting classes', *J. Comput. System Sci.* 48 (1994) 116–148. 86, 92, 92, 93
15. S. FORTUNE, J. HOPCROFT and E. SCHMIDT, 'The complexity of equivalence and containment for free single program schemes', Proceedings of the 5th International Colloquium on Automata, Languages, and Programming, Lecture Notes in Comput. Sci. 62 (ed. G. Ausiello and C. Böhm, Springer-Verlag, 1978) 227–240. 89
16. J. GROLLMANN and A. SELMAN, 'Complexity measures for public-key cryptosystems', *SIAM J. Comput.* 17 (1988) 309–335. 86
17. M. HARRISON, 'Counting theorems and their applications to classification of switching functions', *Recent developments in switching theory*, (ed. A. Mukhopadhyay, Academic Press, 1971) 4–22. 88
18. J. HARTMANIS and Y. YESHA, 'Computation times of NP sets of different densities', *Theoret. Comput. Sci.* 34 (1984) 17–32. 90
19. L. HEMASPAANDRA and J. ROTHE, 'Unambiguous computation: Boolean hierarchies and sparse Turing-complete sets', *SIAM J. Comput.* 26 (1997) 634–653. 86

20. L. HEMASPAANDRA and J. ROTHE, ‘A second step towards complexity-theoretic analogs of Rice’s theorem’, *Theoret. Comput. Sci.* To appear. 86, 91
21. L. HEMASPAANDRA, S. JAIN and N. VERESHCHAGIN, ‘Banishing robust Turing completeness’, *Internat. J. Found. Comput. Sci.* 4 (1993) 245–265. 86
22. C. HOFFMANN, *Group-theoretic algorithms and graph isomorphism*, Lecture Notes in Comput. Sci. 136 (Springer-Verlag, 1982). 90
23. J. KÖBLER, U. SCHÖNING, S. TODA and J. TORÁN, ‘Turing machines with few accepting computations and low sets for PP’, *J. Comput. System Sci.* 44 (1992) 272–286. 86, 90, 93, 93
24. J. KÖBLER, U. SCHÖNING and J. TORÁN, *The graph isomorphism problem: its structural complexity* (Birkhauser, 1993). 90, 90
25. E. LUKS, ‘Isomorphism of graphs of bounded valence can be tested in polynomial time’, *J. Comput. System Sci.* 25 (1982) 42–65. 90
26. R. RAO, J. ROTHE and O. WATANABE, ‘Upward separation for FewP and related classes’, *Inform. Process. Lett.* 52 (1994) 175–180. 86
27. U. SCHÖNING, ‘Probabilistic complexity classes and lowness’, *J. Comput. System Sci.* 39 (1989) 84–100. 88
28. Y. TAKENAGA, M. NOUZOE and S. YAJIMA, ‘Size and variable ordering of OBDDs representing threshold functions’, Proceedings of the 3rd Annual International Computing and Combinatorics Conference, Lecture Notes in Comput. Sci. 1276 (ed. T. Jiang and D. T. Lee, Springer-Verlag, 1997) 91–100. 89
29. L. VALIANT, ‘The relative complexity of checking and evaluating’, *Inform. Process. Lett.* 5 (1976) 20–23. 86
30. K. WAGNER, ‘The complexity of combinatorial problems with succinct input representations’, *Acta Inform.* 23 (1986) 325–356. 91

Bernd Borchert [bb@math.uni-heidelberg.de](mailto:bb@math.uni-heidelberg.de)

Mathematisches Institut  
Universität Heidelberg  
69120 Heidelberg, Germany

Lane A. Hemaspaandra [lane@cs.rochester.edu](mailto:lane@cs.rochester.edu)

Department of Computer Science  
University of Rochester  
Rochester, NY 14627, USA

Jörg Rothe [rothe@informatik.uni-jena.de](mailto:rothe@informatik.uni-jena.de)

Institut für Informatik  
Friedrich-Schiller-Universität Jena  
07740 Jena, Germany