NEW COLLAPSE CONSEQUENCES OF NP HAVING SMALL CIRCUITS

JOHANNES KÖBLER* AND OSAMU WATANABE†

Abstract. We show that if a self-reducible set has polynomial-size circuits, then it is low for the probabilistic class ZPP(NP). As a consequence we get a deeper collapse of the polynomial-time hierarchy PH to ZPP(NP) under the assumption that NP has polynomial-size circuits. This improves on the well-known result of Karp, Lipton, and Sipser (1980) stating a collapse of PH to its second level Σ_2^{P} under the same assumption. As a further consequence, we derive new collapse consequences under the assumption that complexity classes like UP, FewP, and $\mathrm{C_=P}$ have polynomial-size circuits.

Finally, we investigate the circuit-size complexity of several language classes. In particular, we show that for every fixed polynomial s, there is a set in ZPP(NP) which does not have $O(s(n))$-size circuits.

Key words. polynomial-size circuits, advice classes, lowness, randomized computation

AMS subject classifications. 03D10, 03D15, 68Q10, 68Q15

1. Introduction. The question of whether intractable sets can be efficiently decided by non-uniform models of computation has motivated much work in structural complexity theory. In research from the early 1980's to the present, a variety of results has been obtained showing that this is impossible under plausible assumptions (see, e.g., the survey [18]). A typical model for non-uniform computations are circuit families. In the notation of Karp and Lipton [22], sets decidable by polynomial-size circuits are precisely the sets in P/poly, i.e., they are decidable in polynomial time with the help of a polynomial length bounded advice function [32].

Karp and Lipton (together with Sipser) [22] proved that no NP-complete set has polynomial size circuits (in symbols NP $\not\subseteq$ P/poly) unless the polynomial time hierarchy collapses to its second level. The proof given in [22] exploits a certain kind of self-reducibility of the well-known NP-complete problem SAT. More generally, it is shown in [8, 7] that every (Turing) self-reducible set in P/poly is low for the second level Σ_2^{P} of the polynomial time hierarchy. Intuitively speaking, a set is low for a relativizable complexity class if it gives no additional power when used as an oracle for that class.

In this paper, we show that every self-reducible set in P/poly is even low for the probabilistic class ZPP(NP), meaning that ZPP(NP(A)) = ZPP(NP). Since for every oracle A, $\Sigma_2^{\mathrm{P}}(A) = \exists \cdot \mathrm{ZPP}(\mathrm{NP}(A))$, lowness for ZPP(NP) implies lowness for Σ_2^{P}. As a consequence of our lowness result we get a deeper collapse of the polynomial-time hierarchy to ZPP(NP) under the assumption that NP has polynomial-size circuits. At

*Abteilung für Theoretische Informatik, Universität Ulm, Oberer Eselsberg, D-89069 Ulm, Germany (`koebler@informatik.uni-ulm.de`).

† Department of Computer Science, Tokyo Institute of Technology, Meguro-ku, Tokyo 152, Japan (`watanabe@cs.titech.ac.jp`). Part of this work has been done while visiting the University of Ulm (supported in part by the guest scientific program of the University of Ulm).

least in some relativized world, the new collapse level is quite close to optimal: there is an oracle relative to which NP is contained in P/poly but PH does not collapse to P(NP) [17, 39].

We also derive new collapse consequences from the assumption that complexity classes like UP, FewP, and $C_{=}P$ have polynomial-size circuits. Furthermore, our lowness result implies new *relativizable* collapses for the case that $\mathrm{Mod}_m\mathrm{P}$, PSPACE, or EXP have polynomial-size circuits. As a final application, we derive new circuit-size lower bounds. In particular, it is shown (by relativizing proof techniques) that for every fixed polynomial s, there is a set in ZPP(NP) which does not have $O(s(n))$-size circuits. This improves on the result of Kannan [21] that for every polynomial s, the class $\Sigma_2^{\mathrm{P}} \cap \Pi_2^{\mathrm{P}}$ contains such a set. It further follows that in every relativized world, there exist sets in the class ZPEXP(NP) that do not have polynomial-size circuits. It should be noted that there is a non-relativizing proof for a stronger result. As a corollary to the result in [4], which is proved by a non-relativizing technique, it is provable that $\mathrm{MA}_{\mathrm{exp}} \cap \text{co-}\mathrm{MA}_{\mathrm{exp}}$ (a subclass of ZPEXP(NP)) contains non P/poly sets [12, 36].

Some explanation of how our work builds on prior techniques is in order. The proof of our lowness result heavily uses the universal hashing technique [13, 34] and builds on ideas from [2, 14, 24]. For the design of a *zero error* probabilistic algorithm which, with the help of an NP oracle, simulates a given ZPP(NP(A)) computation (where A is a self-reducible set in P/poly) we further make use of the newly defined concept of half-collisions. More precisely, we show how to compute on input 0^n in expected polynomial time a hash family H that can be used to decide all instances of A of length up to n by a strong NP computation. The way H is used to decide (non)membership to A is by checking whether H leads to a half-collision on certain sets. Very recently, Bshouty, Cleve, Gavaldà, Kannan, and Tamon [11] building on a result from [19] have shown that the class of all circuits is exactly learnable in (randomized) expected polynomial time with equivalence queries and the aid of an NP oracle. This immediately implies that for every set A in P/poly an advice function can be computed in FZPP(NP(A)), i.e., by a probabilistic oracle transducer T in expected polynomial time under an oracle in NP(A). More precisely, since the circuit produced by the probabilistic learning algorithm of [11] depends on the outcome of the coin flips, T computes a *multi-valued* advice function, i.e., on input 0^n, T accepts with probability at least 1/2, and on every accepting path, T outputs some circuit that correctly decides all instances of length n w.r.t. A. Using the technique in [11] we are able to show that every self-reducible set A in P/poly even has an advice function in FZPP(NP). Although this provides a different way to deduce the ZPP(NP) lowness of all self-reducible sets in P/poly, we prefer to give a self-contained proof using the "half-collision technique" that does not rely on the mentioned results in [11, 19].

The paper is organized as follows: Section 2 introduces notation and defines the self-reducibility that we use. In Section 3 we prove the ZPP(NP) lowness of all self-reducible sets in P/poly. In Section 4 we state the collapse consequences, and the new circuit-size lower bounds are derived in Section 5.

2. Preliminaries and notation. All languages are over the binary alphabet $\Sigma = \{0,1\}$. As usual, we denote the lexicographic order on Σ^* by $\leq$. The *length* of a string $x \in \Sigma^*$ is denoted by $|x|$. $\Sigma^{\leq n}$ ($\Sigma^{<n}$) is the set of all strings of length at most n (resp., of length smaller than n). For a language A, $A^{=n} = A \cap \Sigma^n$ and $A^{\leq n} = A \cap \Sigma^{\leq n}$. The cardinality of a finite set A is denoted by $|A|$. The *characteristic function* of A is defined as $A(x) = 1$ if $x \in A$, and $A(x) = 0$, otherwise. For a class $\mathcal{C}$ of sets, co-$\mathcal{C}$ denotes the class $\{\Sigma^* - A \mid A \in \mathcal{C}\}$. To encode pairs (or tuples) of strings we use a standard polynomial-time computable pairing function denoted by $\langle \cdot, \cdot \rangle$ whose inverses are also computable in polynomial time. Where intent is clear we write $f(x_1, \ldots, x_k)$ in place of $f(\langle x_1, \ldots, x_k \rangle)$. $\mathcal{N}$ denotes the set of non-negative integers. Throughout the paper, the base of log is 2.

The textbooks [9, 10, 25, 31, 33] can be consulted for the standard notations used in the paper and for basic results in complexity theory. For definitions of probabilistic complexity classes like ZPP see also [15].

An *NP machine* M is a polynomial-time nondeterministic Turing machine. We assume that each computation path of M on a given input x either accepts, rejects, or outputs "?". *M accepts* on input x, if M performs at least one accepting computation, otherwise *M rejects x*. *M strongly accepts (strongly rejects) x* [26] if

- there is at least one accepting (resp., rejecting) computation path and
- there are no rejecting (resp., accepting) computation paths.

If M strongly accepts or strongly rejects x, M is said to perform a *strong computation* on input x. An NP machine that on every input performs a strong computation is called a *strong NP machine*. It is well known that exactly the sets in NP $\cap$ co-NP are accepted by strong NP machines [26].

Next we define the kind of self-reducibility that we use in this paper.

DEFINITION 2.1. *Let $\succ$ be an irreflexive and transitive order relation on Σ^*. A sequence $x_0, x_1, \ldots, x_k$ of strings is called a $\succ$-chain (of length k) from x_0 to x_k if $x_0 \succ x_1 \succ \cdots \succ x_k$. Relation $\succ$ is called* length checkable *if there is a polynomial q such that*

1. *for all $x, y \in \Sigma^*$, $x \succ y$ implies $|y| \leq q(|x|)$,*
2. *the language $\{\langle x, y, k \rangle \mid$ there is a $\succ$-chain of length k from x to $y\}$ is in* NP.

DEFINITION 2.2. *A set A is* self-reducible, *if there is a polynomial-time oracle machine M_{self} and a length checkable order relation $\succ$ such that $A = L(M_{self}, A)$ and on any input x, M_{self} queries the oracle only about strings $y \prec x$.*

It is straightforward to check that the polynomially related self-reducible sets introduced by Ko [23] as well as the length-decreasing and word-decreasing self-reducible sets of Balcázar [6] are self-reducible in our sense. Furthermore, it is well-known (see, for example, [9, 6, 29]) that complexity classes like NP, Σ_k^{P}, Π_k^{P}, PP, $\mathrm{C_{=}P}$, $\mathrm{Mod}_m\mathrm{P}$, PSPACE, and EXP have many-one complete self-reducible sets.

Karp and Lipton [22] introduced the notion of advice functions in order to characterize non-uniform complexity classes. A function $h : \mathcal{N} \to \Sigma^*$ is called a *polynomial-length function* if for some polynomial p and for all $n \geq 0$, $|h(n)| = p(n)$. For a class $\mathcal{C}$ of sets, let $\mathcal{C}$/poly be the class of sets A such that there is a set $I \in \mathcal{C}$ and a polynomial-length function h such that for all n and for all $x \in \Sigma^{\leq n}$,

$$x \in A \Leftrightarrow \langle x, h(n) \rangle \in I.$$

Function h is called an *advice function* for A, whereas I is the corresponding *interpreter set*.

In this paper we will heavily make use of the "hashing technique" which has been very fruitful in complexity theory. Here we review some notations and facts about hash families. We also extend the notion of "collision" by introducing the concept of a "half-collision" which is central to our proof technique.

Sipser [34] used universal hashing, originally invented by Carter and Wegman [13], to decide (probabilistically) whether a finite set X is large or small. A linear hash function h from Σ^m to Σ^k is given by a Boolean (k,m)-matrix (a_{ij}) and maps any string $x = x_1 \ldots x_m$ to a string $y = y_1 \ldots y_k$, where y_i is the inner product $a_i \cdot x = \sum_{j=1}^m a_{ij} x_j \pmod 2$ of the i-th row a_i and x.

Let $x \in \Sigma^m$, $Y \subseteq \Sigma^m$, and let h be a linear hash function from Σ^m to Σ^k. Then we say that x *has a collision on* Y *w.r.t.* h if there exists a string $y \in Y$, different from x, such that $h(x) = h(y)$. More generally, if X is a subset of Σ^m and H is a family[1] $(h_1, \ldots, h_l)$ of linear hash functions from Σ^m to Σ^k, then we say that X *has a collision on* Y *w.r.t.* H (*Collision*(X,Y,H) for short) if there is some $x \in X$ that has a collision on Y w.r.t. every h_i in H. That is,

$$\begin{aligned} \mathit{Collision}(X,Y,H) \quad \Leftrightarrow \quad & \exists\, x \in X\, \exists\, y_1, \ldots, y_l \in Y : x \notin \{y_1, \ldots y_l\} \\ & \text{and for all } i = 1, \ldots, l : h_i(x) = h_i(y_i). \end{aligned}$$

If X has a collision on itself w.r.t. H, we simply say that X *has a collision w.r.t.* H. Next we extend the notion of "collision" in the following way. For any X and $Y \subseteq \Sigma^m$, and any family $H = (h_1, \ldots, h_l)$ of linear hash functions, we say that X *has a half-collision on* Y *w.r.t.* H (*Half-Collision*(X,Y,H) for short) if there is some $x \in X$ that has a collision on Y w.r.t. at least $\lceil l/2 \rceil$ many of the hash functions h_i in H. That is,

$$\begin{aligned} \mathit{Half\text{-}Collision}(X,Y,H) \quad \Leftrightarrow \quad & \exists\, x \in X\, \exists\, y_1, \ldots, y_l \in Y : x \notin \{y_1, \ldots y_l\} \\ & \text{and } |\{i \mid 1 \leq i \leq l, h_i(x) = h_i(y_i)\}| \geq \lceil l/2 \rceil. \end{aligned}$$

An important relationship between collisions and half-collisions is the following one: If X has a collision w.r.t. H on $Y = Y_1 \cup Y_2$, then X must have a half-collision w.r.t. H either on Y_1 or on Y_2.

Note that the predicate *Collision*(X,Y,H) can be decided in NP provided that membership in X and Y can be tested in NP. More precisely, the language $\{\langle v, H \rangle \mid \mathit{Collision}(X_v, Y_v, H)\}$ (as well as the set $\{\langle v, H \rangle \mid \mathit{Half\text{-}Collision}(X_v, Y_v, H)\}$) belongs to NP, if the sets X_v and Y_v are succinctly represented in such a way that the languages $\{\langle x, v \rangle \mid x \in X_v\}$ and $\{\langle y, v \rangle \mid y \in Y_v\}$ are in NP.

[1] i.e., multiset

We denote the set of all families $H = (h_1, \ldots, h_l)$ of l linear hash functions from Σ^m to Σ^k by $\mathcal{H}(l, m, k)$. The following theorem is proved by a pigeon-hole argument. It says that every sufficiently large set must have a collision w.r.t. any hash family.

THEOREM 2.3. [34] *For any hash family $H \in \mathcal{H}(l, m, k)$ and any set $X \subseteq \Sigma^m$ of cardinality $|X| > l \cdot 2^k$, X must have a collision w.r.t. H.*

On the other hand, we get from the next theorem (called Coding Lemma in [34]) an upper bound on the collision probability for sufficiently small sets.

THEOREM 2.4. [34] *Let $X \subseteq \Sigma^m$ be a set of cardinality at most 2^{k-1}. If we choose a hash family H uniformly at random from $\mathcal{H}(k, m, k)$, then the probability that X has a collision w.r.t. H is at most $1/2$.*

We will also make use of the following extension of Theorem 2.4 which can be proved along the same lines.

THEOREM 2.5. *Let $X \subseteq \Sigma^m$ be a set of cardinality at most 2^{k-s}. If we choose a hash family H uniformly at random from $\mathcal{H}(l, m, k)$, then the probability that X has a collision w.r.t. H is at most $2^{k-s(l+1)}$.*

Gavaldà [14] extended Sipser's Coding Lemma (Theorem 2.4) to the case of a collection $\mathcal{C}$ of exponentially many sets. The following theorem has a similar flavor.

THEOREM 2.6. *Let $\mathcal{C}$ be a collection of at most 2^n subsets of Σ^m, each of which has cardinality at most 2^{k-s}. If we choose a hash family H uniformly at random from $\mathcal{H}(l, m, k)$, then the probability that some $X \in \mathcal{C}$ has a collision w.r.t. H is at most $2^{n+k-s(l+1)}$.*

Proof. By Theorem 2.5, we have that for every fixed $X \in \mathcal{C}$, the probability that it has a collision w.r.t. a randomly chosen hash family $H \in \mathcal{H}(l, m, k)$ is at most $2^{k-s(l+1)}$. Hence, the probability that there exists such a set $X \in \mathcal{C}$ is at most $2^{n+k-s(l+1)}$. □

In this paper we make use of a corresponding result for the case of half-collisions.

THEOREM 2.7. *Let $X \subseteq \Sigma^m$ and let $\mathcal{C}$ be a collection of at most 2^n subsets of Σ^m, each of which has cardinality at most 2^{k-s-2}. If we choose a hash family H uniformly at random from $\mathcal{H}(l, m, k)$, then the probability that X has a half-collision on some $Y \in \mathcal{C}$ w.r.t. H is at most $|X| \cdot 2^{n-sl/2}$.*

Proof. For every fixed $Y \in \mathcal{C}$ and every fixed $x \in X$, the probability that x has a collision on Y w.r.t. a randomly chosen h is at most 2^{-s-2}. Hence, the probability that x has a collision on Y w.r.t. at least half of the functions in a randomly chosen hash family $H \in \mathcal{H}(l, m, k)$ is at most

$$\sum_{i=\lceil l/2 \rceil}^{l} \binom{l}{i} (2^{-s-2})^i (1 - 2^{-s-2})^{l-i} \leq 2^{-(s+2)l/2} \sum_{i=\lceil l/2 \rceil}^{l} \binom{l}{i} \leq 2^{l-(s+2)l/2} = 2^{-sl/2}.$$

That is, the probability that x has a half-collision on Y w.r.t. a randomly chosen hash family H is bounded by $2^{-sl/2}$. Hence, the probability that there exists a $Y \in \mathcal{C}$ and an $x \in X$ such that x has a half-collision on Y w.r.t. H is at most $|X| \cdot 2^{n-sl/2}$. □

3. Lowness of self-reducible sets in P/poly. In this section, we show that every self-reducible set A in (NP ∩ co-NP)/poly is low for ZPP(NP). Let $I \in$ NP ∩ co-NP be an interpreter set and h be an advice function for A. We construct a probabilistic algorithm T and an NP oracle O having the following two properties:

a) The expected running time of T is polynomially bounded.

b) On every computation path on input 0^n, T with oracle O outputs some information that can be used to determine the membership to A of any x up to length n by some strong NP computation (in the sense of [26]).

Using these properties, we can prove the lowness of A for ZPP(NP) as follows: In order to simulate any NP(A) computation, we first precompute the above mentioned information for A (up to some length) by T^O, and then by using this information, we can simulate the NP(A) computation by some NP(NP $\cap$ co-NP) computation. Note that the precomputation (performed by T^O) can be done in ZPP(NP), and since NP(NP $\cap$ co-NP) = NP, the remaining computation can be done in NP. Hence, NP(A) $\subseteq$ ZPP(NP), which implies further that ZPP(NP(A)) $\subseteq$ ZPP(ZPP(NP)) (= ZPP(NP) [41]).

We will now make the term "information" precise. For this, we need some additional notation. Let the self-reducibility of A be witnessed by a polynomial-time oracle machine M_{self}, a length checkable order relation $\succ$, and a polynomial q. We assume that $|h(q(n))| = p(n)$ for some fixed polynomial $p > 0$. In the following, we fix n and consider instances of length up to $q(n)$ as well as advice strings of length exactly $p(n)$.

- A *sample* is a sequence $\langle x_1, b_1\rangle\#\cdots\#\langle x_k, b_k\rangle$ of pairs, where the x_i's are instances of length up to $q(n)$ and $b_i = A(x_i)$ for $i = 1, \ldots, k$.
- For any sample $S = \langle x_1, b_1\rangle\#\cdots\#\langle x_k, b_k\rangle$, let $\mathit{Consistent}(S)$ be the set of all advice strings w that are consistent with S, i.e.

$$\mathit{Consistent}(S) = \{w \in \Sigma^{p(n)} \mid \forall i\,(1 \le i \le k) : I(x_i, w) = b_i\}.$$

  The cardinality of $\mathit{Consistent}(S)$ is denoted by $c(S)$.
- For any sample S and any instance x, let $\mathit{Accept}(x, S)$ (resp., $\mathit{Reject}(x, S)$) be the set of all consistent advice strings that *accept* x (resp., *reject* x):

$$\mathit{Accept}(x, S) = \{w \in \mathit{Consistent}(S) \mid I(x, w) = 1\}$$

  and

$$\mathit{Reject}(x, S) = \{w \in \mathit{Consistent}(S) \mid I(x, w) = 0\}.$$

- Let $\mathit{Correct}(x, S)$ be the set $\{w \in \mathit{Consistent}(S) \mid I(x, w) = A(x)\}$ of consistent advice strings that decide x correctly, and let $\mathit{Incorrect}(x, S)$ be the complementary set $\{w \in \mathit{Consistent}(S) \mid I(x, w) \neq A(x)\}$.

Note that the sets $\mathit{Accept}(x, S)$ and $\mathit{Reject}(x, S)$ (as well as $\mathit{Correct}(x, S)$ and $\mathit{Incorrect}(x, S)$) form a partition of the set $\mathit{Consistent}(S)$, and that

$$\begin{array}{lcl} x \in A & \Rightarrow & \mathit{Correct}(x, S) = \mathit{Accept}(x, S) \text{ and } \mathit{Incorrect}(x, S) = \mathit{Reject}(x, S), \\ x \notin A & \Rightarrow & \mathit{Correct}(x, S) = \mathit{Reject}(x, S) \text{ and } \mathit{Incorrect}(x, S) = \mathit{Accept}(x, S). \end{array}$$

The above condition b) can now be precisely stated as follows:

b) On every computation path on input 0^n, T^O outputs a pair $\langle S, H\rangle$ consisting of a sample S and a linear hash family H such that for all x up to length n, $\mathit{Consistent}(S)$ has a half-collision w.r.t. H on $\mathit{Correct}(x,S)$, but not on $\mathit{Incorrect}(x,S)$.

Once we have a pair $\langle S, H\rangle$ satisfying condition b), we can determine whether an instance x of length up to n is in A by simply checking whether $\mathit{Consistent}(x,S)$ has a half-collision w.r.t. H on $\mathit{Accept}(x,S)$ or on $\mathit{Reject}(x,S)$. Since condition b) guarantees that the half-collision can always be found, this checking can be done by a strong NP computation. Let us now prove our main lemma.

LEMMA 3.1. *For any self-reducible set A in* $(\mathrm{NP}\cap\text{co-NP})/\mathrm{poly}$, *there exist a probabilistic transducer T and an oracle O in NP satisfying the above two conditions.*

Proof. We use the notation introduced so far. Recall that $q(n)$ is a length bound on the queries occuring in the self-reduction tree produced by M_{self} on any instance of length n and that $p(n)$ is the advice length for the set of all instances of length up to $q(n)$. Let l be the polynomial defined as $l(n) = 2(q(n)+p(n)+1)$. Further, we denote by $\Sigma^{\preceq n}$ the set $\{y \mid \exists x \in \Sigma^{\leq n}, y \preceq x\}$. Then it is clear that $\Sigma^{\leq n} \subseteq \Sigma^{\preceq n} \subseteq \Sigma^{\leq q(n)}$. A description of T is given below.

input 0^n
$S := \emptyset$
loop
  for $k = 1, \ldots, p(n)$, choose H_k randomly from $\mathcal{H}(l(n), p(n), k)$,
  $k_{max} := \max\{k \mid \mathit{Consistent}(S) \text{ has a collision w.r.t. } H_k\}$
  if there exists an $x \in \Sigma^{\preceq n}$ such that $\mathit{Consistent}(S)$ has
    a half-collision on $\mathit{Incorrect}(x,S)$ w.r.t. $H_{k_{max}}$
  then
    use oracle O to find such a string x and to determine $A(x)$
    $S := S\#\langle x, A(x)\rangle$
  else exit(loop) end
end loop
output $\langle S, H_{k_{max}}\rangle$

Starting with the empty sample, T enters the main loop. During each execution of the loop, T first randomly guesses a series of $p(n)$ many hash families $H_k \in \mathcal{H}(l(n), p(n), k), 1 \leq k \leq p(n)$. Then T computes the integer k_{max} as the maximum $k \in \{1, \ldots, p(n)\}$ such that $\mathit{Consistent}(S)$ has a collision w.r.t. H_k. Notice that by a padding trick we can assume that $c(S)$ is always larger than $2l(n)$, implying that $\mathit{Consistent}(S)$ must have a collision w.r.t. H_1. Since, in particular, $\mathit{Consistent}(S)$ has a collision w.r.t. $H_{k_{max}}$, it follows that for every instance $x \in \Sigma^{\preceq n}$, $\mathit{Consistent}(S)$ has a half-collision w.r.t. $H_{k_{max}}$ on either $\mathit{Correct}(x,S)$ or $\mathit{Incorrect}(x,S)$. If there exists a string $x \in \Sigma^{\preceq n}$ such that $\mathit{Consistent}(S)$ has a half-collision on $\mathit{Incorrect}(x,S)$ w.r.t. $H_{k_{max}}$, then this string is added to the sample S and T continues executing the loop. (We will describe below how T uses the NP oracle O to find x in this case.) Otherwise, the pair $\langle S, H_{k_{max}}\rangle$ fulfills the properties stated in condition b) and T halts.

We now show that the expected running time of T is polynomially bounded. Since the initial size of *Consistent*(S) is $2^{p(n)}$, and since *Consistent*(S) never becomes empty, it suffices to prove that for some polynomial r, T eliminates in each single execution of the main loop with probability at least $1/r(n)$ at least an $1/r(n)$-fraction of the circuits in *Consistent*(S). In fact, we will show that each single extension of S by a pair $\langle x, A(x)\rangle$ reduces the size of *Consistent*(S) with probability at least $1-2^{-l(n)}$ by a factor smaller than $1-1/2^7l(n)$. Since T can only perform more than $2^7l(n)p(n)$ loop iterations, if during some iteration of the main loop T extends S by a pair $\langle x, A(x)\rangle$ which does not shrink the size of *Consistent*(S) by a factor smaller than $1-1/2^7l(n)$, the probability for this event is bounded by $2^7l(n)p(n)\cdot 2^{-l(n)} = o(1)$.

Let S be a sample and let k_{max} be the corresponding integer as determined by T during some specific execution of the loop. We first derive a lower bound for k_{max}. Let k_0 be the smallest integer $k \geq 1$ such that $c(S) \leq l(n)2^{k+1}$. Since either $k_{max} = p(n)$ or *Consistent*(S) does not have a collision w.r.t. the hash family $H_{k_{max}+1} \in \mathcal{H}(l(n), p(n), k_{max}+1)$, we have (using Theorem 2.3) that $c(S) \leq l(n)2^{k_{max}+1}$. Hence, $k_{max} \geq k_0$.

Since T expands S only by strings $x \in \Sigma^{\preceq n}$ such that *Consistent*(S) has a half-collision on *Incorrect*(x, S) w.r.t. $H_{k_{max}}$, and since *Consistent*$(S\#\langle x, A(x)\rangle)$ = *Consistent*(S) − *Incorrect*(x, S), the probability that the size of *Consistent*(S) does not decrease by a factor smaller than $1-1/2^7l(n)$ is bounded by the probability that, w.r.t. $H_{k_{max}}$, *Consistent*(S) has a half-collision on some set *Incorrect*(x, S) of size at most $c(S)/2^7l(n)$. Let

$$\mathcal{C} = \{\mathit{Incorrect}(x, S) \mid x \in \Sigma^{\preceq n}, |\mathit{Incorrect}(x, S)| \leq c(S)/2^7l(n)\}.$$

Since $|\mathcal{C}| < 2^{q(n)+1}$ and since $c(S)/2^7l(n) \leq 2^{k_0-6} = 2^{k_0+k-(k+4)-2}$, it follows from Theorem 2.7 that the probability of *Consistent*(S) having a half-collision on some $Y \in \mathcal{C}$ w.r.t. a uniformly at random chosen hash family $H \in \mathcal{H}(l(n), p(n), k_0+k)$ is at most

$$\begin{aligned} c(S)\cdot 2^{q(n)+1-(k+4)l(n)} &\leq 2^{p(n)+q(n)+1-(k+4)l(n)/2} \\ &= 2^{-(k+3)l(n)/2}. \end{aligned}$$

Thus the probability that for some $k \geq 0$, *Consistent*(S) has a half-collision w.r.t. H_{k_0+k} on some set *Incorrect*(x, S) which is of size at most $c(S)/2^7l(n)$ is bounded by $\sum_{k\geq 0} 2^{-(k+3)l(n)/2} \leq 2^{-l(n)}$.

We finally show how T determines an instance $x \in \Sigma^{\preceq n}$ (if it exists) such that *Consistent*(S) has a half-collision on *Incorrect*(x, S) w.r.t. $H_{k_{max}}$. Intuitively, we use the self-reducibility of A to test the "correctness" w.r.t. A of the "program" $\langle S, H_{k_{max}}\rangle$, where we say that

- a pair $\langle S, H\rangle$ accepts an instance x if *Consistent*(S) has a half-collision on *Accept*(x, S) w.r.t. H, and
- $\langle S, H\rangle$ rejects x if *Consistent*(S) has a half-collision on *Reject*(x, S) w.r.t. H.

Notice that an (incorrect) program might accept and at the same time reject an instance. The main idea to find out whether $\langle S, H_{k_{max}}\rangle$ is incorrect on some instance $x \in \Sigma^{\preceq n}$ (meaning that w.r.t. $H_{k_{max}}$ *Consistent*(S) has a half-collision on *Incorrect*(x, S)) is to test whether the program $\langle S, H_{k_{max}}\rangle$ is in accordance with the output of M_{self} when the oracle queries of M_{self} are answered according to the program $\langle S, H_{k_{max}}\rangle$. To be more precise, consider the NP set

$B = \{\langle z, S, H\rangle \mid$ there is a computation path π of M_{self} on input z fulfilling the following properties:
- if a query q is answered 'yes', then $\langle S, H\rangle$ accepts q,
- if a query q is answered 'no', then $\langle S, H\rangle$ rejects q,
- if π is accepting, then $\langle S, H\rangle$ rejects z, and
- if π is rejecting, then $\langle S, H\rangle$ accepts $z\,\}$.

Then, as shown by the next claim, the correctness of $\langle S, H_{k_{max}}\rangle$ on an instance z can be decided by asking whether $\langle z, S, H_{k_{max}}\rangle$ belongs to B, provided that $\langle S, H_{k_{max}}\rangle$ is correct on all potential queries of M_{self} on input z.

CLAIM. *Assume that $\langle S, H_{k_{max}}\rangle$ is correct on all $y \prec z$. Then $\langle S, H_{k_{max}}\rangle$ is incorrect on z if and only if $\langle z, S, H_{k_{max}}\rangle$ belongs to B.*

Proof. Using the fact that for every instance $x \in \Sigma^{\preceq n}$, *Consistent*(S) has a half-collision w.r.t. $H_{k_{max}}$ on either *Correct*(x, S) or *Incorrect*(x, S), it is easy to see that if $\langle S, H_{k_{max}}\rangle$ is incorrect on z, then the computation path π followed by $M_{self}(z)$ under oracle A witnesses $\langle z, S, H_{k_{max}}\rangle \in B$. For the converse, assume that $\langle z, S, H_{k_{max}}\rangle$ belongs to B and let π be a computation path witnessing this fact. Note that all queries q on π are answered correctly w.r.t. A, since otherwise $\langle S, H_{k_{max}}\rangle$ were incorrect on $q \prec z$. Hence, π is the path followed by $M_{self}(z)$ under oracle A and therefore decides z correctly. On the other hand, since π witnesses $\langle z, S, H_{k_{max}}\rangle \in B$, $\langle S, H_{k_{max}}\rangle$ indeed is incorrect on z. □

Now we can define the oracle set O as $C \oplus D$, where

$C = \{\langle 0^n, x, k, S, H\rangle \mid$ there is a $\succ$-chain of length (at least) k from some string $y \in \Sigma^{\leq n}$ to some string $z \leq x$ such that $\langle z, S, H\rangle \in B\,\}$

and

$D = \{\langle x, S, H\rangle \mid$ there is an accepting computation path π of M_{self} on input x such that any query q is only answered 'yes' ('no') if *Consistent*(S) has a half-collision on *Accept*(q, S) (resp., *Reject*(q, S)) w.r.t. $H\,\}$.

Note that the proof of the claim above also shows that for any $z \in \Sigma^{\preceq n}$ such that $\langle S, H_{k_{max}}\rangle$ is correct on all $y \prec z$, $z \in A$ if and only if $\langle z, S, H_{k_{max}}\rangle$ belongs to D. Now we can complete the description of T. T first asks whether the string $\langle 0^n, 1^{q(n)}, 0, S, H_{k_{max}}\rangle$ belongs to C. It is clear that a negative answer implies that $\langle S, H_{k_{max}}\rangle$ is correct on $\Sigma^{\preceq n}$. Otherwise, by asking queries of the form $\langle 0^n, 1^{q(n)}, i, S, H_{k_{max}}\rangle$, T computes by binary search i_{max} as the maximum value $i \leq 2^{q(n)+1}$ such that $\langle 0^n, 1^{q(n)}, i, 0, S, H_{k_{max}}\rangle$ belongs to C (a similar idea is used

input 0^n
$S := \emptyset$
loop
  for $k = 1, \ldots, p(n)$, choose H_k randomly from $\mathcal{H}(l(n), p(n), k)$,
  $k_{max} := \max\{k \mid \mathit{Consistent}(S)$ has a collision w.r.t. $H_k\}$
  if $\langle 0^n, 1^{q(n)}, 0, S, H_{k_{max}} \rangle \in C$ **then**
    $i_{max} := \max\{i \mid \langle 0^n, 1^{q(n)}, i, 0, S, H_{k_{max}} \rangle \in C\}$
    $x_{min} := \min\{x \mid \langle 0^n, x, i_{max}, 0, S, H_{k_{max}} \rangle \in C\}$
    $S := S\#\langle x, D(x_{min}, S, H_{k_{max}}) \rangle$
  else exit(**loop**) **end**
end loop
output $\langle S, H_{k_{max}} \rangle$

in [27]). Knowing i_{max}, T then determines the lexicographically smallest string x_{min} such that $\langle 0^n, x_{min}, i_{max}, 0, S, H_{k_{max}} \rangle$ is in C. Since $\langle q, S, H_{k_{max}} \rangle \notin C$ holds for all instances $q \prec x_{min}$, it follows inductively from the claim that $\langle S, H_{k_{max}} \rangle$ is correct on all $q \prec x_{min}$. Hence, $\langle S, H_{k_{max}} \rangle$ must be incorrect on x_{min}, and furthermore, T can determine the membership of x_{min} to A by asking whether the string $\langle x_{min}, S, H_{k_{max}} \rangle$ belongs to D. □

THEOREM 3.2. *Every self-reducible set A in the class* $(\mathrm{NP} \cap \text{co-NP})/\mathrm{poly}$ *is low for* $\mathrm{ZPP(NP)}$.

Proof. We first show that $\mathrm{NP}(A) \subseteq \mathrm{ZPP(NP)}$. Let L be a set in $\mathrm{NP}(A)$, and let M be a deterministic polynomial-time oracle machine such that for some polynomial t,

$$L = \{x \mid \exists y \in \Sigma^{t(|x|)} : \langle x, y \rangle \in L(M, A)\}.$$

Let $s(n)$ be a polynomial bounding the length of all oracle queries of M on some input $\langle x, y \rangle$ where x is of length n. Then L can be accepted by a probabilistic oracle machine N using the following NP oracle

$O' = \{\langle x, S, H \rangle \mid$ there is a $y \in \Sigma^{t(|x|)}$ such that M on input $\langle x, y \rangle$ has an accepting path π on which each query q is answered 'yes' ('no') only if $\mathit{Consistent}(S)$ has a half-collision on $\mathit{Accept}(q, S)$ (resp., $\mathit{Reject}(q, S)$) w.r.t. $H\}$.

Here is how N accepts L. On input x, N first simulates T on input $0^{s(|x|)}$ to compute a pair $\langle S, H_{k_{max}} \rangle$ as described above (T asks questions to some NP oracle O). Then N asks the query $\langle x, S, H_{k_{max}} \rangle$ to O' to find out whether x is in L.

This proves that $\mathrm{NP}(A) \subseteq \mathrm{ZPP(NP)}$. Since $\mathrm{ZPP(ZPP)} = \mathrm{ZPP}$ [41] via a proof that relativizes, it follows that $\mathrm{ZPP(NP}(A))$ is also contained in $\mathrm{ZPP(NP)}$, showing that A is low for $\mathrm{ZPP(NP)}$. □

4. Collapse consequences. As a direct consequence of Theorem 3.2 we get an improvement of Karp, Lipton, and Sipser's result [22] that NP is not contained in P/poly unless the polynomial-time hierarchy collapses to Σ_2^{P}.

COROLLARY 4.1. *If* NP *is contained in* $(\mathrm{NP} \cap \text{co-NP})/\mathrm{poly}$ *then the polynomial-time hierarchy collapses to* ZPP(NP).

Proof. Since the NP-complete set SAT is self-reducible, the assumption that NP is contained in $(\mathrm{NP} \cap \text{co-NP})/\mathrm{poly}$ implies that SAT is low for ZPP(NP), and hence the polynomial-time hierarchy collapses to ZPP(NP). ☐

The collapse of the polynomial-time hierarchy deduced in Corollary 4.1 is quite close to optimal, at least in some relativized world [17, 39]: there is an oracle relative to which NP is contained in P/poly but the polynomial-time hierarchy does not collapse to P(NP).

In the rest of this section we report some other interesting collapses which can be easily derived using (by now) standard techniques, and which have also been pointed out independently by several researchers to the second author. First, it is straightforward to check that Theorem 3.2 relativizes: For any oracle B, if A is a self-reducible set in the class $(\mathrm{NP}(B) \cap \text{co-NP}(B))/\mathrm{poly}$, then $\mathrm{NP}(A)$ is contained in $\mathrm{ZPP}(\mathrm{NP}(B))$. Consequently, Theorem 3.2 generalizes to the following result.

THEOREM 4.2. *If A is a self-reducible set in the class* $(\Sigma_k^{\mathrm{P}} \cap \Pi_k^{\mathrm{P}})/\mathrm{poly}$, *then* $\mathrm{NP}(A) \subseteq \mathrm{ZPP}(\Sigma_k^{\mathrm{P}})$.

As a direct consequence of Theorem 4.2 we get an improvement of results in [1, 20] stating (for $k = 1$) that Σ_k^{P} is not contained in $(\Sigma_k^{\mathrm{P}} \cap \Pi_k^{\mathrm{P}})/\mathrm{poly}$ unless the polynomial-time hierarchy collapses to $\Sigma_{k+1}^{\mathrm{P}}$.

COROLLARY 4.3. *Let $k \geq 1$. If Σ_k^{P} is contained in* $(\Sigma_k^{\mathrm{P}} \cap \Pi_k^{\mathrm{P}})/\mathrm{poly}$, *then the polynomial-time hierarchy collapses to* $\mathrm{ZPP}(\Sigma_k^{\mathrm{P}})$.

Proof. Since Σ_k^{P} contains complete self-reducible languages, the assumption that Σ_k^{P} is contained in $(\Sigma_k^{\mathrm{P}} \cap \Pi_k^{\mathrm{P}})/\mathrm{poly}$ implies that $\Sigma_{k+1}^{\mathrm{P}} = \mathrm{NP}(\Sigma_k^{\mathrm{P}}) \subseteq \mathrm{ZPP}(\Sigma_k^{\mathrm{P}})$. ☐

Yap [40] proved that Π_k^{P} is not contained in $\Sigma_k^{\mathrm{P}}/\mathrm{poly}$ unless the polynomial-time hierarchy collapses to $\Sigma_{k+2}^{\mathrm{P}}$. As a further consequence of Theorem 4.2 we get the following improvement of Yap's result.

COROLLARY 4.4. *For $k \geq 1$, if $\Pi_k^{\mathrm{P}} \subseteq \Sigma_k^{\mathrm{P}}/\mathrm{poly}$, then* $\mathrm{PH} = \mathrm{ZPP}(\Sigma_{k+1}^{\mathrm{P}})$.

Proof. The assumption that Π_k^{P} is contained in $\Sigma_k^{\mathrm{P}}/\mathrm{poly}$ implies that $\Sigma_{k+1}^{\mathrm{P}}$ is contained in $\Sigma_k^{\mathrm{P}}/\mathrm{poly} \subseteq (\Sigma_{k+1}^{\mathrm{P}} \cap \Pi_{k+1}^{\mathrm{P}})/\mathrm{poly}$. Hence we can apply Corollary 4.3. ☐

As corollaries to Theorem 4.2, we also have similar collapse results for many other complexity classes. What follows are some typical examples.

COROLLARY 4.5. *For $\mathcal{K} \in \{\mathrm{UP}, \mathrm{FewP}\}$, if $\mathcal{K} \subseteq (\mathrm{NP} \cap \text{co-NP})/\mathrm{poly}$ then $\mathcal{K}$ is low for* ZPP(NP).

Proof. It is well-known that for every set A in UP (FewP), the left set of A [30] is word-decreasing self-reducible and in UP (resp., FewP). Thus, under the assumption that $\mathrm{UP} \subseteq (\mathrm{NP} \cap \text{co-NP})/\mathrm{poly}$ (resp., $\mathrm{FewP} \subseteq (\mathrm{NP} \cap \text{co-NP})/\mathrm{poly}$) it follows by Theorem 3.2 that the left set of A (and since A is polynomial-time many-one reducible to its left set, also A) is low for ZPP(NP). ☐

COROLLARY 4.6. *For every $k \geq 1$, if $\mathrm{C_{=}P} \subseteq (\Sigma_k^{\mathrm{P}} \cap \Pi_k^{\mathrm{P}})/\mathrm{poly}$ then* $\mathrm{CH} = \mathrm{ZPP}(\Sigma_k^{\mathrm{P}})$.

Proof. First, since $\mathrm{C_{=}P}$ has complete word-decreasing self-reducible languages [29], $\mathrm{C_{=}P} \subseteq (\Sigma_k^{\mathrm{P}} \cap \Pi_k^{\mathrm{P}})/\mathrm{poly}$ implies $\mathrm{C_{=}P} \subseteq \mathrm{ZPP}(\Sigma_k^{\mathrm{P}}) \subseteq \mathrm{PH}$. Second, since $\mathrm{PH} \subseteq$

BPP($\mathrm{C_{=}P}$) [37, 35], $\mathrm{C_{=}P} \subseteq (\Sigma_k^{\mathrm{P}} \cap \Pi_k^{\mathrm{P}})/\mathrm{poly}$ implies $\mathrm{PH} \subseteq (\Sigma_k^{\mathrm{P}} \cap \Pi_k^{\mathrm{P}})/\mathrm{poly}$ and therefore PH collapses to $\mathrm{ZPP}(\Sigma_k^{\mathrm{P}})$ by Corollary 4.3. Finally, since $\mathrm{C_{=}P(PH)} \subseteq \mathrm{BPP(C_{=}P)}$ [37], it follows that $\mathrm{C_{=}P(PH)} \subseteq \mathrm{PH}$, and since $\mathrm{CH} = \mathrm{C_{=}P} \cup \mathrm{C_{=}P(C_{=}P)} \cup \ldots$ (see [38]), we get inductively that $\mathrm{CH} \subseteq \mathrm{PH}$ ($\subseteq \mathrm{ZPP}(\Sigma_k^{\mathrm{P}})$). □

COROLLARY 4.7. *Let* $\mathcal{K} \in \{\mathrm{EXP}, \mathrm{PSPACE}, \mathrm{Mod}_m\mathrm{P}\}$, $m \geq 2$. *If for some* $k \geq 1$, $\mathcal{K} \subseteq (\Sigma_k^{\mathrm{P}} \cap \Pi_k^{\mathrm{P}})/\mathrm{poly}$, *then* $\mathcal{K} \subseteq \mathrm{PH}$ *and* PH *collapses to* $\mathrm{ZPP}(\Sigma_k^{\mathrm{P}})$.

Proof. The proof for $\mathcal{K} \in \{\mathrm{EXP}, \mathrm{PSPACE}\}$ is immediate from Theorem 4.2 since PSPACE has complete (length-decreasing) self-reducible languages, and since EXP has complete (word-decreasing) self-reducible languages [6].

The proof for $\mathcal{K} \in \{\mathrm{Mod}_m\mathrm{P} \mid m \geq 2\}$ is analogous to the one of Corollary 4.6 using the fact that $\mathrm{Mod}_m\mathrm{P}$ has complete word-decreasing self-reducible languages [29], and that $\mathrm{PH} \subseteq \mathrm{BPP}(\mathrm{Mod}_m\mathrm{P})$ [37, 35]. □

Since our proof technique is relativizable, the above results hold for every relativized world. On the other hand, it is known that for some classes stronger collapse consequences can be obtained by using non-relativizable arguments.

THEOREM 4.8. [28, 4, 3] *For* $\mathcal{K} \in \{\mathrm{PP}, \mathrm{Mod}_m\mathrm{P}, \mathrm{PSPACE}, \mathrm{EXP}\}$, *if* $\mathcal{K} \subseteq \mathrm{P/poly}$ *then* $\mathcal{K} \subseteq \mathrm{MA}$.

Harry Buhrman pointed out to us that Corollary 4.7 can also be derived from Theorem 4.8.

5. Circuit complexity. Kannan [21] proved that for every fixed polynomial s, there is a set in $\Sigma_2^{\mathrm{P}} \cap \Pi_2^{\mathrm{P}}$ which cannot be decided by circuits of size $s(n)$. Using a padding argument, he obtained the existence of sets in NEXP(NP) ∩ co-NEXP(NP) not having polynomial-size circuits.

THEOREM 5.1. [21]

1. *For every polynomial* s, *there is a set in* $\Sigma_2^{\mathrm{P}} \cap \Pi_2^{\mathrm{P}}$ *that does not have circuits of size* $s(n)$.
2. *For every increasing time-constructible super-polynomial function* $f(n)$, *there is a set in* $\mathrm{NTIME}[f(n)](\mathrm{NP}) \cap \text{co-}\mathrm{NTIME}[f(n)](\mathrm{NP})$ *that does not have polynomial size circuits.*

As an application of our results in Section 3, we can improve Kannan's results in every relativized world from the class $\Sigma_2^{\mathrm{P}} \cap \Pi_2^{\mathrm{P}}$ to ZPP(NP), and from the class $\mathrm{NTIME}[f(n)](\mathrm{NP}) \cap \text{co-}\mathrm{NTIME}[f(n)](\mathrm{NP})$ to $\mathrm{ZPTIME}[f(n)](\mathrm{NP})$, respectively. Here $\mathrm{ZPTIME}[f(n)](\mathrm{NP})$ denotes the class of all sets that are accepted by some zero error probabilistic machine in expected running time $O(f(n))$ relative to some NP oracle.

Note that for all sets in the class P/poly we may fix the interpreter set to some appropriate one in P. Let I_{univ} denote such a fixed interpreter set. Furthermore, P/poly remains the same class, if we relax the notion of an advice function h (w.r.t. I_{univ}) as follows: For every x, $A(x) = I_{univ}(x, h(|x|))$, i.e., $h(n)$ has to decide correctly only $A^{=n}$ (instead of $A^{\leq n}$).

A sequence of circuits C_n, $n \geq 0$, is called a circuit family for A, if for every $n \geq 0$, C_n has n input gates, and for all n-bit strings $x_1 \cdots x_n$, $C_n(x_1, \ldots, x_n) = A(x_1 \cdots x_n)$. It is well-known (see, e.g., [9]) that I_{univ} can be chosen in such a way that advice length and circuit size (i.e., number of gates) are polynomially related to each other. More precisely, we can assume that there is a polynomial p such that the following holds for every set A.

- If h is an advice function for A w.r.t. I_{univ}, then there exists a circuit family C_n, $n \geq 0$, for A of size $|C_n| \leq p(n + |h(n)|)$.
- If C_n, $n \geq 0$, is a circuit family for A, then there exists an advice function h for A w.r.t. I_{univ} of length $|h(n)| \leq p(|C_n|)$.

Moreover, we can assume that for every polynomial-time interpreter set I there is a constant c_I such that if h is an advice function for A w.r.t. I, then there exists an advice function h' for A w.r.t. I_{univ} of length $|h'(n)| \leq |h(n)| + c_I$ for all n.

The following lemma is obtained by a direct diagonalization (cf. the corresponding result in [21]). A set S is called $\mathrm{P}^{\mathcal{C}}$-printable (see [16]) if there is a polynomial-time oracle transducer T and an oracle set $A \in \mathcal{C}$ such that on any input 0^n, T^A outputs a list of all strings in $S^{\leq n}$.

LEMMA 5.2. *For every fixed polynomial s, there is a Δ_3^{P}-printable set A such that every advice function h for A is of length $|h(n)| \geq s(n)$, for almost all n.*

Proof. For a given n, let $x_1, x_2, \ldots, x_{2^n}$ be the sequence of strings of length n, enumerated in lexicographic order. Consider the two sets *Have-Advice* and *Find-A* defined as follows:

$$\langle n, a_1 \cdots a_{s(n)} \rangle \in \textit{Have-Advice} \iff \exists\, w \in \Sigma^{<s(n)}, \forall\, i, 1 \leq i \leq \min(s(n), 2^n) : a_i = I_{univ}(x_i, w),$$

$$\langle n, a_1 \cdots a_j 10^{s(n)-j} \rangle \in \textit{Find-A} \iff \exists\, a_{j+1} \cdots a_{s(n)} : \langle n, a_1 \cdots a_j a_{j+1} \cdots a_{s(n)} \rangle \notin \textit{Have-Advice}.$$

Since there are only $2^{s(n)} - 1$ advice strings w in $\Sigma^{<s(n)}$, at least one pair of the form $\langle n, a_1 \cdots a_{s(n)} \rangle$ is not contained in *Have-Advice* (provided that $s(n) \leq 2^n$). Let α_n denote the lexicographically smallest such pair $\langle n, a_1 \cdots a_{s(n)} \rangle$, i.e., there is no advice of length smaller than $s(n)$ that accepts the strings $x_1, \ldots, x_{s(n)}$ according to $a_1, \ldots, a_{s(n)}$.

Define A as the set of all strings x_i ($|x_i| = n$) such that $1 \leq i \leq s(n) \leq 2^n$ and the ith bit of α_n (i.e., a_i) is 1. By a binary search using oracle *Find-A*, α_n is computable in polynomial time. Since *Have-Advice* is in NP and thus *Find-A* is in NP(NP), it follows that A is P(NP(NP))-printable. Since furthermore, for almost all n, $A^{=n}$ has no advice of length smaller than $s(n)$, the lemma follows. □

COROLLARY 5.3. *For every fixed polynomial s, there is a set A in* ZPP(NP) *that does not have circuits of size $s(n)$.*

Proof. If NP does not have polynomial-size circuits, then we can take $A = \mathrm{SAT}$. Otherwise, PH = ZPP(NP) by Corollary 4.1, and thus the theorem easily follows from Lemma 5.2. □

COROLLARY 5.4. *Let f be an increasing, time-constructible, super-polynomial function. Then* ZPTIME$[f(n)]$(NP) *contains a set A that does not have polynomial-size circuits.*

Proof. If NP does not have polynomial-size circuits, then we can take $A = \mathrm{SAT}$. Otherwise, $\mathrm{PH} = \mathrm{ZPP(NP)}$ by Corollary 4.1, and thus it follows from Lemma 5.2 that there is a set B in $\mathrm{ZPTIME}[n^k](\mathrm{NP})$ such that every advice function h for B is of length $|h(n)| \geq n$ for almost all n. By the proof technique of Lemma 5.2, we can assume that in all length n strings of B, 1's only occur at the $O(\log n)$ rightmost positions. Now consider the following set (where n denotes $|x|$)

$$A = \{x \mid 0^{\lfloor f(n)^{1/k} \rfloor - n} x \in B\},$$

and the interpreter set

$$I = \{\langle 0^{\lfloor f(n)^{1/k} \rfloor - n} x, w\rangle \mid \langle x, w\rangle \in I_{univ}\}.$$

Clearly, A belongs to $\mathrm{ZPTIME}[f(n)](\mathrm{NP})$ and I belongs to P. Furthermore, if h is an advice function for A, then we have for every y of the form $0^{\lfloor f(n)^{1/k} \rfloor - n} x$, $|x| = n$, that

$$\begin{aligned} y \in B \quad &\Leftrightarrow \quad \langle y, h(n)\rangle \in I \\ &\Leftrightarrow \quad \langle y, h'(n)\rangle \in I_{univ}, \end{aligned}$$

where $h'(n)$ is a suitable advice function of length $|h'(n)| \leq |h(n)| + c_I$. Thus, it follows for almost all n that

$$|h(n)| \geq |h'(n)| - c_I \geq |y| - c_I = \lfloor f(n)^{1/k} \rfloor - c_I.$$

This shows that the length of h is super-polynomial. ☐

COROLLARY 5.5. *In every relativized world,* ZPEXP(NP) *contains sets that do not have polynomial-size circuits.*

We remark that the above results are proved by relativizable arguments. On the other hand, Harry Buhrman [12] and independently Thomas Thierauf [36] pointed out to us that Theorem 4.8 (which is proved by a non-relativizable proof technique) can be used to show that $\mathrm{MA_{exp}} \cap \text{co-}\mathrm{MA_{exp}}$ contains non P/poly sets. Here, $\mathrm{MA_{exp}}$ denotes the exponential-time version of Babai's class MA [5]. That is, $\mathrm{MA_{exp}} = \mathrm{MA}[2^{n^{O(1)}}]$, where a language L is in $\mathrm{MA}[f(n)]$, if there exists a set $B \in \mathrm{DTIME}[O(n)]$ such that for all x of length n,

$$\begin{aligned} x \in L \quad &\Rightarrow \quad \exists y, |y| = f(n) : \Pr[\langle x, y, z\rangle \in B] > 2/3, \\ x \notin L \quad &\Rightarrow \quad \forall y, |y| = f(n) : \Pr[\langle x, y, z\rangle \in B] < 1/3, \end{aligned}$$

where z is chosen uniformly at random from $\Sigma^{f(n)}$.

COROLLARY 5.6. [12, 36] $\mathrm{MA_{exp}} \cap \text{co-}\mathrm{MA_{exp}}$ *contains sets that do not have polynomial size circuits.*

Since there exist recursive oracles relative to which all sets in EXP(NP) have polynomial size circuits [39, 17], it is not possible to extend Corollary 5.5 by relativizing techniques to the class EXP(NP).

6. Concluding remarks. An interesting question concerning complexity classes $\mathcal{C}$ that are known to be not contained in P/poly but are not known to have complete sets is whether the existence of sets in $\mathcal{C} - \mathrm{P/poly}$ can be *constructively* shown. For example, by Corollary 5.5 we know that the class ZPEXP(NP) contains sets that do not have polynomial-size circuits. But we were not able to give a *constructive* proof of this fact. To the best of our knowledge, no explicit set is known even in $\mathrm{NEXP(NP)} \cap \mathrm{co\text{-}NEXP(NP)} - \mathrm{P/poly}$.

Acknowledgments

For helpful discussions and suggestions regarding this work we are very grateful to H. Buhrman, R. Gavaldà, L. Hemaspaandra, M. Ogihara, U. Schöning, R. Schuler, and T. Thierauf. We like to thank H. Buhrman, L. Hemaspaandra, and M. Ogihara for permitting us to include their observations in the paper.

REFERENCES

[1] M. Abadi, J. Feigenbaum, and J. Kilian, *On hiding information from an oracle*, Journal of Computer and System Sciences, 39 (1989), pp. 21–30.

[2] D. Angluin, *Queries and concept learning*, Machine Learning, 2 (1988), pp. 319–342.

[3] L. Babai and L. Fortnow, *Arithmetization: A new method in structural complexity*, Computational Complexity, 1 (1991), pp. 41–66.

[4] L. Babai, L. Fortnow, and C. Lund, *Non-deterministic exponential time has two-prover interactive protocols*, Computational Complexity, 1 (1991), pp. 1–40.

[5] 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 (1988), pp. 254–276.

[6] J. Balcázar, *Self-reducibility*, Journal of Computer and System Sciences, 41 (1990), pp. 367–388.

[7] J. Balcázar, R. Book, and U. Schöning, *The polynomial-time hierarchy and sparse oracles*, Journal of the ACM, 33 (1986), pp. 603–617.

[8] ———, *Sparse sets, lowness and highness*, SIAM Journal on Computing, 23 (1986), pp. 679–688.

[9] J. Balcázar, J. Díaz, and J. Gabarró, *Structural Complexity Theory I*, Springer-Verlag, second ed., 1995.

[10] D. Bovet and P. Crescenzi, *Introduction to the Theory of Complexity*, Prentice-Hall, 1994.

[11] N. Bshouty, R. Cleve, R. Gavaldà, S. Kannan, and C. Tamon, *Oracles and queries that are sufficient for exact learning*, Tech. Report TR 95-015, ECCC, 1995. To appear in Journal of Computer and System Sciences.

[12] H. Buhrman, *Personal communication*, 1994.

[13] J. L. Carter and M. N. Wegman, *Universal classes of hash functions*, Journal of Computer and System Sciences, 18 (1979), pp. 143–154.

[14] R. Gavaldà, *Bounding the complexity of advice functions*, Journal of Computer and System Sciences, 50 (1995), pp. 468–475.

[15] J. Gill, *Computational complexity of probabilistic complexity classes*, SIAM Journal on Computing, 6 (1977), pp. 675–695.

[16] J. Hartmanis and Y. Yesha, *Computation times of NP sets of different densities*, Theoretical Computer Science, 34 (1984), pp. 17–32.

[17] H. Heller, *On relativized exponential and probabilistic complexity classes*, Information and Control, 71 (1986), pp. 231–243.

[18] L. Hemachandra, M. Ogiwara, and O. Watanabe, *How hard are sparse sets?*, in Proceedings of the 7th Structure in Complexity Theory Conference, IEEE Computer Society Press, 1992, pp. 222–238.

[19] M. Jerrum, L. Valiant, and V. Vazirani, *Random generation of combinatorial structures from a uniform distribution*, Theoretical Computer Science, 43 (1986), pp. 169–188.

[20] J. Kämper, *Non-uniform proof systems: A new framework to describe non-uniform and probabilistic complexity classes*, Theoretical Computer Science, 85 (1991), pp. 305–331.

[21] R. Kannan, *Circuit-size lower bounds and non-reducibility to sparse sets*, Information and Control, 55 (1982), pp. 40–56.

[22] R. M. Karp and R. J. Lipton, *Some connections between nonuniform and uniform complexity classes*, in Proceedings of the 12th ACM Symposium on Theory of Computing, ACM Press, 1980, pp. 302–309.

[23] K. Ko, *On self-reducibility and weak p-selectivity*, Journal of Computer and System Sciences, 26 (1983), pp. 209–221.

[24] J. Köbler, *Locating P/poly optimally in the extended low hierarchy*, Theoretical Computer Science, 134 (1994), pp. 263–285.

[25] J. Köbler, U. Schöning, and J. Torán, *The Graph Isomorphism Problem: Its Structural Complexity*, Birkhäuser, Boston, 1993.

[26] T. Long, *Strong nondeterministic polynomial-time reducibilities*, Theoretical Computer Science, 21 (1982), pp. 1–25.

[27] A. Lozano and J. Torán, *Self-reducible sets of small density*, Mathematical Systems Theory, 24 (1991), pp. 83–100.

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

[29] M. Ogiwara and A. Lozano, *On sparse hard sets for counting classes*, Theoretical Computer Science, 112 (1993), pp. 255–275.

[30] M. Ogiwara and O. Watanabe, *On polynomial-time bounded truth-table reducibility of NP sets to sparse sets*, SIAM Journal on Computing, 20 (1991), pp. 471–483.

[31] C. Papadimitriou, *Computational Complexity*, Addison-Wesley, 1994.

[32] N. Pippenger, *On simultaneous resource bounds*, in Proceedings of the 20th IEEE Symposium on the Foundations of Computer Science, IEEE Computer Society Press, 1979, pp. 307–311.

[33] U. Schöning, *Complexity and Structure*, vol. 211 of Lecture Notes in Computer Science, Springer-Verlag, 1986.

[34] M. Sipser, *A complexity theoretic approach to randomness*, in Proceedings of the 15th ACM Symposium on Theory of Computing, ACM Press, 1983, pp. 330–335.

[35] J. Tarui, *Probabilistic polynomials, AC^0 functions, and the polynomial time hierarchy*, Theoretical Computer Science, 113 (1993), pp. 167–183.

[36] T. Thierauf, *Personal communication*, 1994.

[37] S. Toda and M. Ogiwara, *Counting classes are at least as hard as the polynomial-time hierarchy*, SIAM Journal on Computing, 21 (1992), pp. 316–328.

[38] J. Torán, *Complexity classes defined by counting quantifiers*, Journal of the ACM, 38 (1991), pp. 753–774.

[39] C. Wilson, *Relativized circuit complexity*, Journal of Computer and System Sciences, 31 (1985), pp. 169–181.

[40] C. Yap, *Some consequences of non-uniform conditions on uniform classes*, Theoretical Computer Science, 26 (1983), pp. 287–300.

[41] S. Zachos, *Robustness of probabilistic computational complexity classes under definitional perturbations*, Information and Control, 54 (1982), pp. 143–154.