Electronic Colloquium on Computational Complexity, Report No. 33 (1999)

Graph Isomorphism is Low for ZPP^{NP} and other Lowness results

V. Arvind* Johannes Köbler†

August 19, 1999

Abstract

We show the following new lowness results for the probabilistic class ZPP^{NP}.

- The class $\text{AM} \cap \text{coAM}$ is low for ZPP^{NP}. As a consequence it follows that Graph Isomorphism and several group-theoretic problems known to be in $\text{AM} \cap \text{coAM}$ are low for ZPP^{NP}.
- The class IP[P/poly], consisting of sets that have interactive proof systems with honest provers in P/poly are also low for ZPP^{NP}.

We consider lowness properties of nonuniform function classes: NPMV/poly, NPSV/poly, NPMV_t/poly, and NPSV_t/poly. Specifically, we show that

- Sets whose characteristic functions are in NPSV/poly and that have program checkers (in the sense of Blum and Kannan [8]) are low for AM and ZPP^{NP}.
- Sets whose characteristic functions are in NPMV_t/poly are low for Σ_2^p.

1 Introduction

In the recent past the probabilistic class ZPP^{NP} has appeared in different results and contexts in complexity theory research. E.g. consider the result $\text{MA} \subseteq \text{ZPP}^{\text{NP}}$ [1, 12] which sharpens and improves Sipser's theorem $\text{BPP} \subseteq \Sigma_2^p$. The proof in [1] uses derandomization techniques based on hardness assumptions [21]. Another example is the result that if $\text{SAT} \in \text{P/poly}$ then $\text{PH} = \text{ZPP}^{\text{NP}}$. [19, 4], which improves the classic Karp-Lipton theorem[1] Actually [19] prove that every self-reducible set[2] A in $(\text{NP} \cap \text{co-NP})/\text{poly}$ is *low* for ZPP^{NP}, i.e. $\text{ZPP}^{\text{NP}^A} = \text{ZPP}^{\text{NP}}$. This stronger result is in a sense natural, since there is usually an underlying lowness result that implies a collapse consequence result like the Karp-Lipton theorem. Recall, for example, that the lowness result underlying the Karp-Lipton theorem is that self-reducible sets in P/poly are low for Σ_2^p [24].

The notion of lowness was first introduced in complexity theory by Schöning [24]. It has since then been an important conceptual tool in complexity theory, see e.g. the survey paper [15].

*Institute of Mathematical Sciences, C.I.T Campus, Chennai, India 600 113 (arvind@imsc.ernet.in)

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

[1]The Karp-Lipton theorem states that if $\text{SAT} \in \text{P/poly}$ then PH collapses to Σ_2^p.

[2]By self-reducibility we mean word-decreasing self-reducibility which is adequate because standard complexity classes contained in EXP have such self-reducible complete problems.

ISSN 1433-8092

1.1 *Lowness for* $\mathrm{ZPP}^{\mathrm{NP}}$

We recall the formal definition of lowness [24]. For a relativizable complexity class $\mathcal{C}$ such that for all sets A $A \in \mathcal{C}^A$, let *Low*($\mathcal{C}$) denote $\{A \mid \mathcal{C}^A = \mathcal{C}\}$. Clearly, *Low*($\mathcal{C}$) is contained in $\mathcal{C}$ and consists of languages that are powerless as oracle for $\mathcal{C}$.

Few complexity classes have their low sets exactly characterized. These are the well-known examples: $Low(\mathrm{NP}) = \mathrm{NP} \cap \text{co-NP}$, $Low(\mathrm{AM}) = \mathrm{AM} \cap \mathrm{coAM}$ [24]. For most complexity classes however, a complete characterization of low sets appears to be a challenging open question. Regarding $Low(\Sigma_2^p)$, Schöning proved [25] that $\mathrm{AM} \cap \mathrm{coAM}$ is contained in $Low(\Sigma_2^p)$, implying that $Low(\mathrm{AM}) \subseteq Low(\Sigma_2^p)$. This containment is anomalous because $\mathrm{AM} \not\subseteq \Sigma_2^p$ in some relativized worlds [23]. Indeed, lowness appears to have other anomalous properties: it is not known to preserve containment of complexity classes, for example $\mathrm{NP} \subseteq \mathrm{PP}$ but $\mathrm{NP} \cap \text{co-NP}$ is not known to be in *Low*(PP). Similarly, $\mathrm{NP} \subseteq \mathrm{MA}$ but $\mathrm{NP} \cap \text{co-NP}$ is not known to be in *Low*(MA). Little is known about *Low*(MA) except that it contains BPP and is contained in $\mathrm{MA} \cap \text{co-MA}$ [17].

Regarding $\mathrm{ZPP}^{\mathrm{NP}}$, it is shown in [19] that $Low(\mathrm{ZPP}^{\mathrm{NP}}) \subseteq Low(\Sigma_2^p)$. No characterization of $Low(\mathrm{ZPP}^{\mathrm{NP}})$ is known. Our aim is to show some inclusions in $Low(\mathrm{ZPP}^{\mathrm{NP}})$ as a first step.

We first show in this paper that $\mathrm{AM} \cap \mathrm{coAM}$ is low for $\mathrm{ZPP}^{\mathrm{NP}}$, i.e. $\mathrm{AM} \cap \mathrm{coAM} \subseteq Low(\mathrm{ZPP}^{\mathrm{NP}})$. Hence we have the inclusion chain: $Low(\mathrm{MA}) \subseteq Low(\mathrm{AM}) \subseteq Low(\mathrm{ZPP}^{\mathrm{NP}}) \subseteq Low(\Sigma_2^p)$. It follows that Graph Isomorphism and other group-theoretic problems known to be in $\mathrm{AM} \cap \mathrm{coAM}$ [3] are low for $\mathrm{ZPP}^{\mathrm{NP}}$.

We prove another lowness result for $\mathrm{ZPP}^{\mathrm{NP}}$: Let IP[P/poly] denote languages that have interactive proof systems with honest prover in P/poly. We show that $\mathrm{IP[P/poly]} \subseteq Low(\mathrm{ZPP}^{\mathrm{NP}})$, improving the containment $\mathrm{IP[P/poly]} \subseteq Low(\Sigma_2^p)$ shown in [2]. Our proof has a derandomization component in which the Nisan-Wigderson pseudorandom generator [21] is used to derandomize the verifier in the IP[P/poly] protocol. The rest of the proof is based on the random sampling technique as applied in [4, 16].

1.2 $\mathrm{NP/poly} \cap \text{co-NP/poly}$ *and subclasses*

As shown in [19], lowness proofs that work for P/poly carry over easily to $(\mathrm{NP} \cap \text{co-NP})/\mathrm{poly}$. However there are technical hurdles in handling $\mathrm{NP/poly} \cap \text{co-NP/poly}$: E.g. the best known collapse consequence of $\mathrm{NP} \subseteq \mathrm{NP/poly} \cap \text{co-NP/poly}$ is $\mathrm{PH} \subseteq \mathrm{ZPP}(\Sigma_2^p)$, and it is just a relativized version of the result in [19].

In order to better understand this aspect of $\mathrm{NP/poly} \cap \text{co-NP/poly}$ the authors of [9]introduce two interesting subclasses of $\mathrm{NP/poly} \cap \text{co-NP/poly}$ which we discuss in Section 5. We notice firstly that $\mathrm{NP/poly} \cap \text{co-NP/poly}$ and the above-mentioned subclasses are closely connected to the function classes NPMV/poly, NPSV/poly, NPMV_t/poly, and NPSV_t/poly, which are nonuniform analogues of the function classes NPMV, NPSV, NPMV_t, and NPSV_t introduced and studied by Selman and other researchers [26, 10]. More precisely, we note that $A \in (\mathrm{NP} \cap \text{co-NP})/\mathrm{poly}$ if and only if $\chi_A \in \mathrm{NPSV}_t/\mathrm{poly}$, where χ_A denotes the characteristic function of a language A. Similarly, $A \in \mathrm{NP/poly} \cap \text{co-NP/poly}$ if and only if $\chi_A \in \mathrm{NPMV/poly}$. Likewise, NPSV/poly and NPMV_t/poly capture the two new subclasses of $\mathrm{NP/poly} \cap \text{co-NP/poly}$ defined in [9].

We prove the following new lowness results for these classes:

- We show that self-reducible sets whose characteristic functions are in NPMV_t/poly are low for Σ_2^p (this result is essentially the lowness result underlying the collapse consequence i.e. Theorem 5.2 in [9]).

- We show that all self-checkable sets[3] whose characteristic functions are in NPSV/poly are low for AM.

2 Preliminaries

Let $\Sigma = \{0,1\}$. We denote the cardinality of a set X by $\|X\|$ and the length of a string $x \in \Sigma^*$ by $|x|$. The characteristic function of a language $L \subseteq \Sigma^*$ is denoted by χ_L. The definitions of standard complexity classes like P, NP, E, EXP etc. can be found in standard books [7, 22]. A relativized complexity class $\mathcal{C}$ with oracle A is denoted by either $\mathcal{C}^A$ or $\mathcal{C}(A)$. Likewise, we denote an oracle Turing machine M with oracle A by M^A or $M(A)$.

For a class $\mathcal{C}$ of sets and a class $\mathcal{F}$ of functions from 1^* to Σ^*, let $\mathcal{C}/\mathcal{F}$ [13] be the class of sets A such that there is a set $B \in \mathcal{C}$ and a function $h \in \mathcal{F}$ such that for all $x \in \Sigma^*$,

$$x \in A \Leftrightarrow \langle x, h(1^{|x|})\rangle \in B.$$

The function h is called an *advice function* for A.

We recall definitions of AM and MA. A language L is in AM if there exist a polynomial p and a set $B \in \mathrm{P}$ such that for all x, $|x| = n$,

$$\begin{aligned} x \in A &\Rightarrow \mathrm{Prob}_{r \in_R \{0,1\}^{p(n)}}[\exists y, |y| = p(n) : \langle x, y, r\rangle \in B] \;=\; 1, \\ x \notin A &\Rightarrow \mathrm{Prob}_{r \in_R \{0,1\}^{p(n)}}[\forall y, |y| = p(n) : \langle x, y, r\rangle \in B] \le 1/4. \end{aligned}$$

Let L be a set in MA. Then there exist a polynomial p and a set $B \in \mathrm{P}$ such that for all x, $|x| = n$,

$$\begin{aligned} x \in A &\Rightarrow \exists y, |y| = p(n) : \mathrm{Prob}_{r \in_R \{0,1\}^{p(n)}}[\langle x, y, r\rangle \in B] \ge 3/4, \\ x \notin A &\Rightarrow \forall y, |y| = p(n) : \mathrm{Prob}_{r \in_R \{0,1\}^{p(n)}}[\langle x, y, r\rangle \in B] \le 1/4. \end{aligned}$$

Notice that we have taken the definition of AM with 1-sided error, known to be equivalent to AM with 2-sided error.

Next, we recall some properties of universal hashing: let $\mathcal{L}(m, k)$ denote all linear functions from Σ^m to Σ^k, where Σ^m and Σ^k are interpreted as m and k-dimensional vector spaces over GF[2], respectively. We recall a useful folklore lemma (as stated in [16]) that lower bounds the probability that a random $h \in \mathcal{L}(m, k)$ *isolates* some x in a given set S of appropriate size (meaning that x is the only element in S such that $h(x) = 0^k$). The lemma also upper bounds the probability that such an x belongs to a given small subset S' of S.

Lemma 1 [16] *Let $S \subseteq \Sigma^m - \{0^m\}$ be a nonempty set of size s, let $S' \subseteq S$ be of size at most $s/6$, and let $k \in \mathcal{N}$ such that $2^k < 3s \le 2^{k+1}$. Then, for $h \in \mathcal{L}(m, k)$ chosen uniformly at random,*

- *with probability at least $2/9$, there is a unique $x \in S$ such that $h(x) = 0^k$, and*
- *with probability at most $1/9$, there exists some $x \in S'$ such that $h(x) = 0^k$.*

Definitions for single and multiprover interactive proof systems can be found in standard texts, e.g. [22]. Let MIP denote the class of languages with multiprover interactive protocols and IP denote the class of languages with single-prover interactive protocols. We denote by MIP[$\mathcal{C}$] and IP[$\mathcal{C}$] the respective language classes where the prover complexity is bounded by FP($\mathcal{C}$).

[3] In the program checking sense of Blum and Kannan [8]

3 $\mathrm{AM} \cap \mathrm{coAM}$ is low for $\mathrm{ZPP}^{\mathrm{NP}}$

In this section we show that $\mathrm{AM} \cap \mathrm{coAM}$ is low for $\mathrm{ZPP}^{\mathrm{NP}}$. It follows that Graph Isomorphism and a host of group-theoretic problems known to be in $\mathrm{AM} \cap \mathrm{coAM}$ [3] are all low for $\mathrm{ZPP}^{\mathrm{NP}}$. We recall here that it is already known that $\mathrm{AM} \cap \mathrm{coAM}$ is low for Σ_2^p [25] and also for AM [17].

We notice first that although $\mathrm{AM} \cap \mathrm{coAM} \subseteq \mathrm{ZPP}^{\mathrm{NP}}$ (because $\mathrm{AM} \subseteq \mathrm{coR}^{\mathrm{NP}}$ and the equality $\mathrm{ZPP} = \mathrm{R} \cap \mathrm{coR}$ relativizes) and $\mathrm{AM} \cap \mathrm{coAM}$ is low for itself, it doesn't follow that $\mathrm{AM} \cap \mathrm{coAM}$ is low for $\mathrm{ZPP}^{\mathrm{NP}}$. As mentioned before, $\mathrm{NP} \cap \text{co-NP}$ is trivially low for NP but is not known to be low for PP or MA.

Theorem 2 $\mathrm{AM} \cap \mathrm{coAM}$ *is low for* $\mathrm{ZPP}^{\mathrm{NP}}$.

Proof. Let L be any set in $\mathrm{AM} \cap \mathrm{coAM}$. We need to show that a given $\mathrm{ZPP}^{\mathrm{NP}^L}$ machine M can be simulated in $\mathrm{ZPP}^{\mathrm{NP}}$. For an input x of length bounded by n to the machine M suppose all the lengths of the queries made to L during the computation are bounded by m. Since $L \in \mathrm{AM} \cap \mathrm{coAM}$, it follows from standard probability amplification techniques and quantifier swapping that there are NP sets A and B and a polynomial p such that $\forall y : |y| \leq m$, there is a subset $S \subseteq \{0,1\}^{p(m)}$ of size $||S|| \geq 2^{p(m)-1}$ with the following property:

$y \in L$ implies

$$\forall w : \langle y, w \rangle \in A \text{ and } \forall w \in S : \langle y, w \rangle \notin B$$

and $y \notin L$ implies

$$\forall w : \langle y, w \rangle \in B \text{ and } \forall w \in S : \langle y, w \rangle \notin A$$

Notice that in the above we are using the fact that AM protocols can be assumed to have one-sided error.

In other words, a large fraction of the w's act as advice strings using which membership in L for strings of length m can be decided with an $\mathrm{NP} \cap \text{co-NP}$ computation. Notice, however, that it would be incorrect for us to claim from here that $L \in (\mathrm{NP} \cap \text{co-NP})/\text{poly}$, because if we use a string from $\{0,1\}^{p(m)} - S$ as advice, the resulting combination of machines for A and B may not yield an $\mathrm{NP} \cap \text{co-NP}$ computation for some input $y \in \Sigma^{\leq m}$. However, we observe that the above property of advice strings in S implies that $w \in S$ iff using w as advice yields an $\mathrm{NP} \cap \text{co-NP}$ computation for all inputs $y \in \Sigma^{\leq m}$.

Thus, a candidate advice $w \in \Sigma^{p(m)}$ is *not* in S iff it satisfies the following NP predicate:

$$\exists y \in \Sigma^{\leq m} : \langle y, w \rangle \in A \cap B$$

We now describe the $\mathrm{ZPP}^{\mathrm{NP}}$ machine N that simulates the given $\mathrm{ZPP}^{\mathrm{NP}^L}$ machine M on some input x. Machine N first randomly guesses an advice string in $w \in \Sigma^{p(m)}$ which, by assumption, is in S with probability 1/2. A single NP query using the above NP predicate is now used to certify that $w \in S$. Using such a w as advice, N can replace the oracle L with an $\mathrm{NP} \cap \text{co-NP}$ computation when it simulates M. ■

Corollary 3 *Graph Isomorphism is low for* $\mathrm{ZPP}^{\mathrm{NP}}$.

The above corollary follows since Graph Isomorphism is in $\mathrm{AM} \cap \mathrm{coAM}$ [11]. The lowness result also holds for various group-theoretic problems known to be in $\mathrm{AM} \cap \mathrm{coAM}$ [3].

Notice that the previous theorem essentially shows that we can simulate AM $\cap$ coAM with an NP $\cap$ co-NP computation using a random string in a coNP set as advice for the computation. This observation combined with the result of [19] (that self-reducible sets in (NP $\cap$ co-NP)/poly are low for ZPP$^{\rm NP}$) immediately yields the following corollary.

Corollary 4 *Self-reducible sets in* AM $\cap$ coAM/poly *are low for* ZPP$^{\rm NP}$.

Additionally, we also have the following corollary in the average-case complexity setting. We first recall the definition of $\mathcal{AP}$ (see, e.g. [18] for a detailed treatment): $\mathcal{AP}$ is the class of decision problems A such that for every polynomial-time computable distribution there is an algorithm that decides A and is polynomial-time on the average for that distribution.

Corollary 5 *If* NP $\subseteq \mathcal{AP}$ *then* AM $\cap$ coAM = NP $\cap$ co-NP.

The proof follows from the assumption NP $\subseteq \mathcal{AP}$ combined with the fact that for any set in AM $\cap$ coAM a large fraction of strings satisfying a coNP predicate are good advice strings, as we have already seen in the proof of Theorem 2. Thus, a ZPP computation can randomly guess such an advice string and use an AP algorithm for the *uniform* distribution to decide the coNP predicate. This $\mathcal{AP}$ algorithm, with its running time truncated to a suitable polynomial bound, will still accept many of the randomly picked good advice strings.[4]

4 IP[P/poly] is low for ZPP$^{\rm NP}$

The class IP[P/poly] implicitly figures in the proof of the result in [5] that if EXP $\subseteq$ P/poly then EXP = MA. We recall the idea of the proof: if EXP is contained in P/poly, then, each language in EXP has a multiprover interactive protocol in which the provers are in EXP (and hence by assumption have polynomial size circuits). This MIP protocol can be simulated by an MA protocol where Merlin simply sends the circuits for the provers to Arthur in the first round. In other words, the proof shows the inclusion chain EXP $\subseteq$ IP[P/poly] $\subseteq$ MA. Since the MA protocol is a single prover interactive protocol, we also have MIP[P/poly] = IP[P/poly] $\subseteq$ MA.

The above collapse consequence result of [5] motivates the study of lowness properties of IP[P/poly]. We show next that IP[P/poly] $\subseteq$ *Low*(ZPP$^{\rm NP}$), improving the containment IP[P/poly] $\subseteq$ *Low*(Σ_2^p) shown in [2]. Our result strengthens the result of [16] that NP sets in P/poly with self-computable witnesses are low for ZPP$^{\rm NP}$. IP[P/poly] contains such NP sets, but IP[P/poly] may not even be contained in NP. Although IP[P/poly] $\subseteq$ MA $\subseteq$ AM, IP[P/poly] is not known to be closed under complement, and it is not known if IP[P/poly] is contained in AM. Thus, IP[P/poly] $\subseteq$ *Low*(ZPP$^{\rm NP}$) appears incomparable to AM $\cap$ coAM $\subseteq$ *Low*(ZPP$^{\rm NP}$) shown in Theorem 2 in the previous section. Our result is also incomparable to the result in [19] that self-reducible sets in P/poly are low for ZPP$^{\rm NP}$. An interesting aspect of our proof is that it combines derandomization and almost uniform random sampling.

We recall definitions and results on derandomization [21]. For $s \in \mathcal{N}$, $\mathcal{CIR}(n,s)$ denotes all boolean functions $f : \{0,1\}^n \to \{0,1\}$ that can be computed by deterministic circuits of size s. Furthermore, for a function $s : \mathcal{N} \to \mathcal{N}^+$ let $\mathcal{CIR}(s) = \bigcup_{n\geq 0} \mathcal{CIR}(n, s(n))$.

Definition 6 *(cf. [21]) Let $r : \mathcal{N} \to \mathcal{R}^+$ and L be any language. L is said to be* $\mathcal{CIR}(r)$-hard *if for all but finitely many n*

$$\frac{1}{2} - \frac{1}{r(n)} < \frac{\|\{x \in \{0,1\}^n \mid \chi_L(x) = g(x)\}\|}{2^n} < \frac{1}{2} + \frac{1}{r(n)}.$$

[4] This is an application of ideas from [18].

Let p, l, m, k be positive integers. A collection $D = (D_1, \ldots, D_p)$ of sets $D_i \subseteq \{1, \ldots, l\}$ is called a (p, l, m, k)-*design* if $\|D_i\| = m$ for all i, and for all $i \neq j$, $\|D_i \cap D_j\| \leq k$. Using D we get from a boolean function $g : \{0,1\}^m \to \{0,1\}$ a sequence of boolean functions $g_i : \{0,1\}^l \to \{0,1\}$, $i = 1, \ldots, p$, defined as $g_i(s_1, \ldots, s_l) = g(s_{i_1}, \ldots, s_{i_m})$ where $D_i = \{i_1, \ldots, i_m\}$. By concatenating the values of these functions we get a function $g_D : \{0,1\}^l \to \{0,1\}^p$ where $g_D(s) = g_1(s) \ldots g_p(s)$. Nisan and Wigderson show [21, Lemma 2.4] that the output of g_D looks random to a small deterministic circuit, provided g is hard to approximate by deterministic circuits of a certain size (in other words, the hardness of g implies that the pseudorandom generator g_D is secure against small circuits). The following makes this more precise.

Lemma 7 [21] *Let D be a (p, l, m, k)-*design *and let $g : \{0,1\}^m \to \{0,1\}$ be an $\mathcal{CIR}^A(m, p^2 + p2^k)$-hard function. Then the function g_D has the property that for every p-input circuit c of size at most p^2,*

$$\left| \text{Prob}_{y \in_R \{0,1\}^p}[c^A(y) = 1] - \text{Prob}_{s \in_R \{0,1\}^l}[c^A(g_D(s)) = 1] \right| \leq 1/p.$$

Next, we recall the main theorem of [21]:

Theorem 8 [21] *For all $\alpha > 0$ if* E *has a language that is $\mathcal{CIR}(2^{\alpha n})$-hard, then* BPP = P.

We also need the following folklore lemma which states that most boolean functions are hard on the average (see e.g. [20]).

Lemma 9 *For each α such that $0 < \alpha < 1/3$, there is a constant n_0 such that for all $n \geq n_0$ the number of n-ary boolean functions that are not $\mathcal{CIR}(n, 2^{\alpha n})$-hard is at most $2^{2^n} \cdot e^{-2^{n/4}}$.*

Theorem 10 IP[P/poly] *is low for* ZPP^{NP}.

Proof. As observed before each language in IP[P/poly] is in MA via the following protocol: Merlin (the prover) first sends to Arthur (the verifier) a polynomial-size circuit for the honest prover. Arthur uses this circuit to simulate the IP[P/poly] interactive protocol for the given language. This is simply a randomized BPP-like computation. More precisely, for $L \in$ IP[P/poly] there are a polynomial p and a set $A \in$ P such that $\forall n$,

$$\exists w \in \{0,1\}^{p(n)} \forall y \in L^{\leq n} \; : \; \text{Prob}_{r \in_R \{0,1\}^{p(n)}}[\langle y, w, r \rangle \in A] \geq 3/4$$

and

$$\forall w \in \{0,1\}^{p(n)} \forall y \in \Sigma^{\leq n} - L^{\leq n} \; : \; \text{Prob}_{r \in_R \{0,1\}^{p(n)}}[\langle y, w, r \rangle \in A] \leq 1/4$$

For $L \in$ IP[P/poly] we need to show that given a ZPP^{NP^L} machine M there is a ZPP^{NP} machine N that accepts the same language. Let x be a length n_0 input to M. Suppose all queries made to L during the computation of $M(x)$ are of size at most n. In the design of N, we will have two preprocessing steps which are both ZPP^{NP} computations. The preprocessing steps will correctly compute a polynomial-size circuit for $L^{\leq n}$ which can be used to replace the oracle in machine M to complete the proof. For the rest of the proof we fix the input x to machine M.

To proceed further, we use the above MA protocol for L. For a pair y, w, the decision procedure for A can be seen as a circuit $C_{y,w}$ that takes r as input. We can assume w.l.o.g that $C_{y,w}$ as size bounded by $p^2(n)$. Using Lemma 7 we have for any (p, l, m, k)-design D and any $\mathcal{CIR}(m, p^2 + p2^k)$-hard boolean function $g : \{0,1\}^m \to \{0,1\}$ that

$$\left| \text{Prob}_{r \in_R \{0,1\}^p}[c(r) = 1] - \text{Prob}_{s \in_R \{0,1\}^l}[c(g_D(s)) = 1] \right| \leq 1/p$$

holds for every p-input circuit c of size at most p^2. Now let $m(n) = 12 \log p(n)$, $l(n) = 288 \log p(n)$, and $k(n) = \log p(n)$. By Lemma 9 we know that for all sufficiently large n, a randomly chosen boolean function $g : \{0,1\}^{m(n)} \to \{0,1\}$ is $\mathcal{CIR}(m(n), 2^{m(n)/4})$-hard (and thus $\mathcal{CIR}(m(n), p(n)^2 + p(n)2^{k(n)})$-hard) with probability at least $1 - e^{-2^{m(n)/4}}$.

The machine N performs the following preprocessing step:

> **input** y, $|y| \leq n$;
> compute a $(p(n), l(n), m(n), k(n))$-design D;
> **choose randomly** $g : \{0,1\}^{m(n)} \to \{0,1\}$;
> **if** g is $\mathcal{CIR}(m(n), 2^{m(n)/4})$-hard **then** {this can be decided by an NP oracle}
> compute the pseudorandom strings $r_1, \ldots, r_{2^{l(n)}}$ of g_D on all seeds;

By the property of these pseudorandom strings $r_1, \ldots, r_{2^{l(n)}}$ with respect to circuits $C_{y,w}$, we have for all y: $|y| \leq n$ that

$$\exists w \in \{0,1\}^{p(n)} \forall y \in L^{\leq n} \ : \ \|\{r_i | \langle y, w, r_i \rangle \in A\}\| \geq 2^{l(n)-1}$$

and

$$\forall w \in \{0,1\}^{p(n)} \forall y \in \Sigma^{\leq n} - L^{\leq n} \ : \ \|\{r_i | \langle y, w, r_i \rangle \in A\}\| < 2^{l(n)-1}$$

.

For each n, therefore, we can now efficiently build a polynomial-size circuit C_n (in which the pseudorandom strings $r_1, \ldots, r_{2^{l(n)}}$ are hard-wired) such that

$$\exists w \in \{0,1\}^{p(n)} \forall y \in L^{\leq n} \ : \ C_n(y, w) = 1$$

and

$$\forall w \in \{0,1\}^{p(n)} \forall y \in \Sigma^{\leq n} - L^{\leq n} \ : \ C_n(y, w) = 0$$

.

Notice that $\{C_n\}_{n>0}$ is a *uniform* circuit family, where each C_n takes an input y and advice string w to decide y's membership in L. The above property guarantees that the advice strings w have only 1-sided error.

We now proceed to the next step of machine N in which it performs a $\mathrm{ZPP}^{\mathrm{NP}}$ computation to compute with high probability a polynomial-size deterministic circuit $\hat{c}$ that decides L correctly for inputs of length upto n. In fact, each output circuit $\hat{c}_W$ will be constructed from a set W of polynomially many advice strings in $\Sigma^{p(n)}$. Stated formally, for all $x \in \Sigma^n$

$$\hat{c}_W(x) = 1 \iff \exists w \in W \ : \ C_n(w, x) = 1$$

By virtue of the 1-sided correctness of C_n, $\hat{c}_W$ rejects all $x \in \Sigma^n - L^{=n}$ for any W.

We need one more notation: For $S \subseteq L^{=n}$, define the active advice set $W(S)$ to be

$$W(S) := \{w \in \Sigma^{p(n)} \mid \forall x \in S : C_n(w, x) = 1\}$$

Notice that $W(S)$ contains all correct advice strings for any $S \subseteq L^{=n}$.

On input 0^n, machine N iteratively includes strings $x \in L^{=n}$ into S, until it finds a circuit $\hat{c}_W$ for $L^{=n}$. N aims to extend S with an x such that $\|W(S)\|$ decreases by a constant factor. To ensure this, N randomly picks $9n$ hash functions h_i and computes the set W of (at most $9n$) advice strings that are isolated in $W(S)$ by some h_i. Then N includes in S the lexicographically least $x \in L^{=n}$ such that $\forall w \in W : \ C_n(w, x) = 0$. We now formally describe N:

input 0^n;
$S := \emptyset$;
loop
  choose randomly $k \in \{1, \ldots, p(n)\}$;
  choose randomly $h_1, \ldots, h_{9n}$ from $\mathcal{L}(p(n), k)$;
  $W := \{w \in \Sigma^{p(n)} \mid$ some h_i isolates w within $W(S)\}$
  if $\hat{c}_W(x) = 0$ for some $x \in L^{=n}$ **then** $S := S \cup \{x\}$
  else exit(loop) end
end loop;
output $\hat{c}_W$

Clearly, N can be implemented with access to an NP oracle. Moreover, since $\hat{c}_W$ never accepts an $x \notin L^{=n}$ and since the loop only terminates outputing $\hat{c}_W$ when it accepts all $x \in L^{=n}$, the algorithm is correct. It only remains to show that the expected running time of N is polynomially bounded.

Consider a specific stage of the loop iteration. Call $x \in L^{=n}$ a *good* extension of S if $\|W(S)\|$ decreases by a constant factor, say, $\|W(S \cup \{x\})\| < (5/6) \cdot \|W(S)\|$. Let A denote the event that $2^k < 3\|W(S)\| \leq 2^{k+1}$ holds for the randomly picked k. Clearly, A holds with probability $\frac{1}{p(n)}$. We claim that $p_S = \text{Prob}_{h_1,\ldots,h_{9n}}[$ a good extension of S is obtained $\mid A\,] \geq 1/2$.

To see this, let $BAD = \{x \in L^{=n} \mid \|W(S) - W(S \cup \{x\})\| \leq \|W(S)\|/6\}$ denote the set of bad extensions of S. For $x \in BAD$, let p_x be the probability that S is extended by x conditioned on event A. Notice that $1 - p_S \leq \sum_{x \in BAD} p_x$. Now, p_x is bounded by the probability that $\hat{C}_W(x) = 0$ for the set W of isolated advice strings. Note that $\hat{c}_W(x) = 0$ if none of $h_1, h_2, \ldots, h_{9n}$ isolates within $W(S)$ an advice string $w \in W(S \cup \{x\})$. By Lemma 1, a single h_i isolates some advice string in $W(S)$ with probability greater than $2/9$, and, moreover, h_i isolates an advice string in $W(S) - W(S \cup \{x\})$ with probability at most $1/9$. Thus, with probability at least $1/9$, each h_i isolates an advice string in $W(S \cup \{x\})$. So, the probability that none of $h_1, \ldots, h_{9n}$ isolates an advice string in $W(S \cup \{x\})$ is at most $(8/9)^{9n} < e^{-n}$. Hence, $p_x \leq e^{-n}$ for each $x \in BAD$. Since $||L^{=n}|| \leq 2^n$ we get $1 - p_S \leq \sum_{x \in BAD} p_x \leq (2/e)^n$. Thus, $p_S \geq 1/2$ for large enough n. Therefore, the probability that a single extension of S is good is at least $\frac{1}{2p(n)}$ (since $\text{Prob}[A] = \frac{1}{p(n)}$).

The size of $W(S)$ is $2^{p(n)}$ at the start of $N(0^n)$'s computation. Since $W(S)$ is always nonempty, there can be at most $p(n) \log^{-1}(6/5) < 4p(n)$ successful extensions of S. Hence, it follows that the expected number of loop iterations is at most $8p^2(n)$. ■

The above lowness result easily extends to IP[(NP ∩ co-NP)/poly] by observing that the proof relativizes: for any oracle set A, IP[P^A/poly] is low for ZPP^{NP^A}.

We conclude this section with another connection to the average-case complexity setting.

Theorem 11 *If* NP $\subseteq \mathcal{AP}$ *and* NP $\subseteq$ P/poly *then PH collapses to* Δ_2^p.

Proof. Recall from [19, 4] that if NP $\subseteq$ P/poly then PH collapses to ZPP^{NP}. At the heart of this collapse result is the following FZPP^{NP} computation: on input 0^n it outputs with high probability a polynomial-size circuit for length n instances of SAT. Since NP $\subseteq$ P/poly by assumption, the NP oracle in the above FZPP^{NP} computation can be replaced by an appropriate polynomial-size circuit. Thus, given access to a hard boolean function we can use the Nisan-Wigderson generator to derandomize the above FZPP^{NP} computation: Observe that derandomization here implies that the output of the pseudrandom generator will include a pseudorandom string that is an accepting computation of the $\mathrm{FZPP}^{\mathrm{NP}}$ computation. Thus, given access to a hard boolean function the $\mathrm{FZPP}^{\mathrm{NP}}$ computation can be derandomized to an $\mathrm{FP}^{\mathrm{NP}}$ computation.

Now, as argued in the proof of the previous theorem, we can use a $\mathrm{ZPP}^{\mathrm{NP}}$ computation to guess a hard boolean function and then verify that it is hard with a single coNP query. At this point, we can use the assumption that $\mathrm{NP} \subseteq \mathcal{AP}$, as in [18] and Corollary 5, to get rid of the NP oracle and replace this $\mathrm{ZPP}^{\mathrm{NP}}$ computation with an ZPP computation. Finally, notice that the lexicographically first output of this ZPP computation can be computed by an $\mathrm{FP}^{\mathrm{NP}}$ computation. Thus it is possible to compute a polynomial-size circuit for $\mathrm{SAT}^{=n}$ by an $\mathrm{FP}^{\mathrm{NP}}$ computation and consequently PH collapses to Δ_2^p. ∎

5 Nonuniform function classes and lowness

We now study lowness properties of NPMV/poly, NPSV/poly, NPMV_t/poly, and NPSV_t/poly. These are nonuniform analogs of the function classes NPMV, NPSV, NPMV_t, and NPSV_t studied by Selman [26] and other researchers, e.g. [10]. We find these nonuniform classes interesting because when restricted to characteristic functions of sets, NPMV/poly coincides with NP/poly ∩ co-NP/poly and NPSV_t/poly coincides with NP ∩ co-NP/poly. Likewise, we note that the two subclasses of NP/poly ∩ co-NP/poly studied in [9], namely all sets underproductively reducible to sparse sets and all sets overproductively reducible to sparse sets, also coincide with NPSV/poly and NPMV_t/poly, respectively.

Following Selman's notation in [26], a transducer is an NDTM T with a write-only output tape. On input x machine T outputs $y \in \Sigma^*$ if there is an accepting path on input x along which y is output. Hence, the function defined by T on Σ^* could be multivalued and partial. Given a multivalued function f on Σ^* and $x \in \Sigma^*$ we use the notation

$$\textit{set-}f(x) = \{y \mid f(x) \mapsto y\}$$

to denote the (possibly empty) set of functions values for input x. We recall the basic definitions.

Definition 12 [10]

1. NPMV *is the class of multivalued, partial functions f for which there is a polynomial-time NDTM N such that*
   - *(a) $f(x)$ is defined iff $N(x)$ has an accepting path.*
   - *(b) $y \in \textit{set-}f(x)$ if and only if there is an accepting path of $N(x)$ where y is output.*
2. NPSV *is the class of single-valued partial functions in* NPMV.
3. NPMV_t *is the class of total functions in* NPMV.
4. NPSV_t *is the class of total single-valued functions in* NPMV.

The classes NPMV/poly, NPSV/poly, NPMV_t/poly, and NPSV_t/poly are the standard nonuniform analogs of the above classes defined as usual [13]: for $\mathcal{F} \in \{\mathrm{NPMV}, \mathrm{NPSV}, \mathrm{NPMV}_t\mathrm{NPSV}_t\}$, a multivalued partial function f is in $\mathcal{F}$/poly if there is a function $g \in \mathcal{F}$, a polynomial p, and an *advice function* $h : 1^* \mapsto \Sigma^*$ with $|h(1^n)| \leq p(n)$ for all n, such that for all $x \in \Sigma^*$,

$$\textit{set-}f(x) = \textit{set-}g(\langle x, h(1^{|x|})\rangle).$$

Before we connect these classes to NP/poly$\cap$co-NP/poly and its subclasses [9], we recall definitions from [9]: Consider polynomial-time nondeterministic oracle machines N whose computation paths can have three possible outcomes: accept, reject, or **?**. The machine N can also be viewed as a transducer which computes, for given oracle D and input x, a multivalued function. More precisely, if we identify accept with value 1 and reject with 0, and consider the **?** computation paths as rejecting paths then N^D defines a partial multivalued function: *set-*$N^D(x) \subseteq \{0,1\}$. Machine N^D is said to be *underproductive* if for each x we have $\{0,1\} \not\subseteq$ *set-*$N^D(x)$, and N is said to be *robustly underproductive* if for each oracle D and input x we have $\{0,1\} \not\subseteq$ *set-*$N^D(x)$. Likewise, N^D is *overproductive* if for each x we have *set-*$N^D(x) \neq \emptyset$, and N is said to be *robustly overproductive* if for each oracle D and input x we have *set-*$N^D(x) \neq \emptyset$.

With standard arguments we can convert a sparse set into a polynomial-size advice string and vice-versa (see, e.g. [7]). It follows that $A \in$ NP/poly $\cap$ co-NP/poly if and only if there is a sparse set S and a nondeterministic machine N such that N^S is both overproductive and underproductive and $A = L(N^S)$. Similarly, $A \in$ (NP $\cap$ co-NP)/poly if and only if there is a sparse set S and a nondeterministic machine N such that N is both robustly overproductive and robustly underproductive.

Proposition 13 *Let χ_A denote the characteristic function for a set $A \subseteq \Sigma^*$:*

1. *χ_A is in* NPMV/poly *if and only if A is in* NP/poly$\cap$co-NP/poly.
2. *χ_A is in* NPSV$_t$/poly *if and only if A is in* (NP $\cap$ co-NP)/poly.
3. *χ_A is in* NPSV/poly *if and only if there are a sparse set S and a robustly underproductive machine N such that $A = L(N^S)$.*
4. *χ_A is in* NPMV$_t$/poly *if and only if there are a sparse set S and a robustly overproductive machine N such that $A = L(N^S)$.*

By abuse of notation, we identify χ_A with A in this section. E.g. we write $A \in$ NPSV/poly when we mean $\chi_A \in$ NPSV/poly. We now turn to lowness questions for the nonuniform function classes. The classes NP/poly$\cap$co-NP/poly and (NP$\cap$co-NP)/poly are of interest in the context of deriving strong collapse consequences from the assumption that NP (or other hard complexity classes) is contained in one of these classes. We recall the known collapse consequence [19] result for NP/poly$\cap$ co-NP/poly under the assumption that NP is contained therein: If NP $\subseteq$ NP/poly $\cap$ co-NP/poly then PH collapses to ZPP$^{\Sigma_2^p}$. The open question here is whether the collapse consequence can possibly be improved to ZPP$^{\text{NP}}$. This is one reason to consider classes that lie between NP/poly$\cap$ co-NP/poly and (NP $\cap$ co-NP)/poly.

5.1 *A lowness result for* NPMV$_t$/poly

It is shown in [9] that if an NP-complete problem is in NPMV$_t$/poly[5] then PH collapses to Σ_2^p. We use ideas in their proof to show the underlying lowness result for functions: all word-decreasing self-reducible functions in NPMV$_t$/poly are low for Σ_2^p. We first recall the definition of word-decreasing self-reducible sets (and define its obvious extension to total single-valued functions).

Definition 14 [6] *For strings $x, y \in \Sigma^*$, $x \prec y$ if $|x| < |y|$ or $|x| = |y|$ and x is lexicographically smaller than y. A set A is* word-decreasing self-reducible *if there is a polynomial-time oracle*

[5]In [9] the authors state the result in terms of overproductive reductions to sparse sets.

machine M such that $A = L(M^A)$, where on any input x the machine M queries the oracle only about strings y such that $y \prec x$. Similarly, a total single-valued function f on Σ^* is word-decreasing self-reducible *if there is a polynomial-time oracle transducer T such that T^f computes f, where on any input x transducer T can query the oracle only about strings y such that $y \prec x$.*

The definition of lowness extends naturally to total, single-valued functions: A functional oracle f return $f(x)$ on query x. For any relativizable complexity class $\mathcal{C}$ we say that $f \in Low(\mathcal{C})$ if $\mathcal{C}^f = \mathcal{C}$. We show next that self-reducible sets and self-reducible functions in NPMV/poly have identical lowness properties. Hence it suffices to prove lowness of self-reducible sets in NPMV/poly.

Theorem 15 *Let $\mathcal{F}$ contain all self-reducible functions in any of the four function classes $\{\mathrm{NPMV/poly}, \mathrm{NPSV/poly}, \mathrm{NPMV}_t\mathrm{/poly}, \mathrm{NPSV}_t\mathrm{/poly}\}$. Let $\mathcal{C}$ be the subclass of $\mathcal{F}$ consisting of characteristic functions (making $\mathcal{C}$ a language class, essentially). For every self-reducible function $f \in \mathcal{F}$ there is a set $A \in \mathcal{C}$ such that f and A are polynomial-time Turing equivalent.*

Proof. Given $f \in \mathcal{F}$, we can define the corresponding set $A \in \mathcal{C}_{\mathcal{F}}$ by suitably encoding, for each x, the bits of $f(x)$ in A. We can easily ensure that the self-reducibility of f carries over to A and f and A are polynomial-time Turing equivalent. ∎

Theorem 16 *Word-decreasing self-reducible sets in* NPMV_t/poly *are low for* Σ_2^p.

Proof. Let A be a word-decreasing self-reducible set in NPMV_t/poly. Let M_0 be the self-reduction machine for A. Consider a $\Sigma_2^p(A)$ machine M with oracle A. There is a polynomial p such that for inputs x of length n, $p(n)$ bounds the length of the queries made by $M(x)$ to A. We fix n and let m denote $p(n)$. Since $A \in \mathrm{NPMV}_t$/poly there is a nondeterministic transducer T that fulfils conditions of Definition 12. W.l.o.g we assume for each $m \in \mathcal{N}$ that T interprets advice strings $w \in \Sigma^{q(m)}$ for inputs of length at most m, for some polynomial q.

How hard is it to test that a candidate advice w is good? The conjunction of the following two coNP predicates does this task:

- We first define the coNP predicate STRONG(w):

  $$\forall z \in \Sigma^{\leq m}\ \forall y_1, y_2\ :\ \{T(z,w,y_1), T(z,w,y_2)\} \neq \{0,1\}$$

  where $T(z,w,y_1)$ and $T(z,w,y_2)$ are values output by T on computation paths y_1 and y_2, given advice w and input z. Notice that this coNP predicate just verifies that T is single-valued for advice w. However, observe that advice w could still be incorrect. The next coNP predicate checks correctness of w.

- For input $z \in \Sigma^{\leq m}$, let $M_0^w(z)$ denote the computation that results by simulating the self-reduction machine $M_0(z)$, where any query q is answered by simulating $T(w,q)$: the simulation of $T(w,q)$ aborts along a path resulting in **?**. On other paths the simulation proceeds treating output 1 as accept and output 0 as reject. Notice that this simulation of $M_0^w(z)$ yields a nondeterministic *single-valued* computation if STRONG(w) holds, as STRONG(w) forces each simulation of the kind $T(w,q)$ to be single-valued for all q. When the simualation is complete along some path $M_0^w(z)$ accepts on that path and outputs the value computed.

  We now define the coNP predicate CORRECT(w):

CORRECT(w) := $\forall z \in \Sigma^{\leq m}$: if $T(w,z)$ accepts then $M_0^w(z)$ never rejects, and if $T(w,z)$ rejects then $M_0^w(z)$ never accepts.

Notice that if STRONG(w) holds then CORRECT(w) checks that $\forall z \in \Sigma^m$ the advice string w is consistent with the self-reducibility machine M_0.

We have the following claim summarizing the properties of STRONG and CORRECT.

Claim. *The string $w \in \Sigma^{q(m)}$ is a correct advice string for $\Sigma^{\leq m}$ iff* STRONG$(w) \wedge$ CORRECT(w) *holds.*

Continuing with the proof, recall that the computation tree for $M^A(x)$ has an $\exists$ layer followed by a $\forall$ layer. We denote this by saying that $M^A(x)$ accepts if and only if: $\exists y \ : \ M^A(x,y)$, where $M^A(x,y)$ is the remaining computation which defines a co-NP^A predicate. Now it is easy to logically describe the Σ_2^p machine N that simulates M on an input x of length n. N accepts x iff the following Σ_2^p predicate holds:

$$\exists w \in \Sigma^{q(m)} \ \exists y \ : \ M^w(x,y) \ \wedge \ \text{STRONG}(w) \ \wedge \ \text{CORRECT}(w)$$

where $M^w(x,y)$ represents the following computation: simulate $M^A(x,y)$, and for each query q made to A plug in the nondeterministic computation $T(w,q)$. If a computation path of $T(w,q)$ rejects then terminate that computation as accepting (because this path is irrelevant to the overall computation). If a computation path of $T(w,q)$ outputs 1 (interpreted as accept in the simulation) or 0 (interpreted as reject in the simulation), machine $M^w(x,y)$ continues with the computation assuming that the answer is correct. Continuing in this manner $M^w(x,y)$ finally accepts or rejects on each computation path.

To see correctness, notice first that N accepts $x \in \Sigma^n$ if M^A accepts x, because for the good advice string w and for y such that $M^A(x,y)$ is true, $M^w(x,y)$ correctly simulates $M^A(x,y)$ and therefore accepts. Next, for a string $x \in \Sigma^n$ that is rejected by M^A, notice that for the good advice string w, $M^w(x,y)$ also rejects for any y. And for a bad advice string w the coNP predicate STRONG(w) $\wedge$ CORRECT(w) rejects regardless of $M^w(x,y)$ for any y. This completes the proof. ■

Since Σ_k^p, Π_k^p, PP, C$_=$P, Mod$_m$P, PSPACE, and EXP have many-one complete word-decreasing self-reducible sets [6], the following corollary is immediate.

Corollary 17 *If $\mathcal{C} \in \{\Sigma_k^p, \Pi_k^p, \text{PP}, \text{C}_=\text{P}, \text{Mod}_m\text{P}, \text{PSPACE}, \text{EXP}\}$, for $k \geq 1$, has a complete set in* NPMV$_t$/poly *then $\mathcal{C} \subseteq \Sigma_2^p$ and* PH $= \Sigma_2^p$.

The proof follows since for each $\mathcal{C} \in \{\Sigma_k^p, \Pi_k^p, \text{PP}, \text{C}_=\text{P}, \text{Mod}_m\text{P}, \text{PSPACE}, \text{EXP}\}$ we have $\Sigma_3^p \subseteq \Sigma_2^{\mathcal{C}}$.

We end this section with the observation that AM $\cap$ coAM is contained in NPMV$_t$/poly. It in interesting to now compare the lowness results (Theorems 2 and 16) for these classes.

Proposition 18 *If $L \in$ AM $\cap$ coAM then L is in* NPMV$_t$/poly.

Proof. Given $L \in$ AM$\cap$coAM, as already observed in an earlier proof by probability amplification techniques and quantifier swapping, there are NP sets A and B and a polynomial p such that $\forall x : |x| \leq m$, there is a subset $S \subseteq \{0,1\}^{p(m)}$ of size $||S|| \geq 2^{p(m)-1}$ with the following property: $x \in L$ implies

$$\forall w : \langle x, w\rangle \in A \text{ and } \forall w \in S : \langle x, w\rangle \notin B$$

and $x \notin L$ implies

$$\forall w : \langle x, w\rangle \in B \text{ and } \forall w \in S : \langle x, w\rangle \notin A$$

We can combine the NP machines for A and B and build a transducer I that takes pair $\langle x, w\rangle$ as input, where w is the advice string. Observe that S constitutes the (large) fraction of the w's that are correct advice strings using which membership in L for strings of length m can be decided and for such advice strings the transducer I will always yield a single-valued, total computation for all inputs of length m, outputing either 1 or 0 depending on the membership of input x. Notice that the above properties also already imply L is in NPMV_t/poly, because no matter which $w \in \{0,1\}^{p(m)}$ is used as advice, $\langle x, w\rangle$ is either in the NP set A or in the NP set B and so the transducer I always outputs at least one of 0 or 1 for any advice string and any input. ■

5.2 *A lowness result for* NPSV/poly

In [9] it is left as an open problem to discover new lowness (or collapse consequence) results for NPSV/poly. As noted in [9], nothing better is known for NPSV/poly than the collapse consequence result: if SAT is in NPSV/poly then PH collapses to $\mathrm{ZPP}^{\Sigma_2^p}$, which holds even for the larger class NP/poly $\cap$ co-NP/poly [19].

We show that sets in NPSV/poly that are checkable, in the sense of program checking as defined by Blum and Kannan [8], are low for AM and for $\mathrm{ZPP}^{\mathrm{NP}}$. Since $\oplus$P, PP,PSPACE, and EXP have checkable complete problems, it follows that for any of these classes inclusion in NPSV/poly implies its containment in AM$\cap$coAM. This result is proved on the same lines as the Babai et al result [5]: If EXP is contained in P/poly then EXP $\subseteq$ MA.

Recall the definitions of MIP[$\mathcal{C}$] and IP[$\mathcal{C}$] for a class $\mathcal{C}$ of languages. We prove a technical lemma that immediately yields the lowness result.

Lemma 19 *If* $A \in$ NPSV/poly *then* MIP[A] $\subseteq$ AM.

Proof. Let $L \in$ MIP[A] for some set $A \in$ NPSV/poly. Let T be the nondeterministic transducer that witnesses that $A \in$ NPSV/poly. We describe an MAM protocol for L:

1. Let x be an input of length n to the protocol. Let $m = p(n)$, for polynomial p, bound the size of the queries to A made by the verifier during the protocol for inputs of length n.

2. **Merlin** sends advice w of length $q(m)$ to Arthur.

3. **Arthur** sends a polynomial random string r (used for simulating the original IP protocol) to Merlin.

4. **Merlin** sends back the list of successive queries to set A (generated by simulating the original IP protocol with random string r), the list of answers to those queries along with the computation paths of transducer T with advice w that certify the answers to the queries.

5. **Arthur** can verify in polynomial time that Merlin's message is all correct and accept iff the original IP protocol accepts.

By the fact that T computes a single-valued partial function for any advice w, although the verifier is simulating the nondeterministic transducer T, it is guaranteed that each accepting computation path has identical output and hence does identical computation. Thus, what makes the above MAM protocol work is the fact that for any advice w and query q all accepting computation paths of $T(w, q)$ output the same value. So, regardless of which computation paths are sent to Arthur by Merlin in Step 4 of the above protocol, Arthur's decision will be the same. In other words, Arthur's acceptance depends only on the random string r, hence exactly preserving the acceptance probability of the original IP protocol.

Standard techniques can be used to convert the MAM protocol to an AM protocol. This completes the proof. ■

We have as immediate consequence the following lowness result.

Theorem 20 *If L is a checkable set in* NPSV/poly *then* $L \in \mathrm{AM} \cap \mathrm{coAM}$ *and hence low for AM and* $\mathrm{ZPP}^{\mathrm{NP}}$.

Proof. The assumption in the theorem's statement implies that both L and $\overline{L}$ are in MIP[L] by the checker characterization theorem of [8]. Now, applying Lemma 19 yields that both L and $\overline{L}$ are in AM and the result follows. ■

We can derive new collapse consequences as corollary, since $\oplus$P, PP,PSPACE, and EXP have checkable complete problems. It follows that for any of these classes inclusion in NPSV/poly implies its containment in $\mathrm{AM} \cap \mathrm{coAM}$.

Corollary 21 *If any of the classes* $\oplus$P, PP,PSPACE, *and* EXP *is contained in* NPSV/poly *then it is low for* AM *and hence* PH = AM.

Notice that we also have the same lowness for checkable funcitons in NPSV/poly.

Theorem 22 *Checkable functions in* NPSV/poly *are low for AM and* $\mathrm{ZPP}^{\mathrm{NP}}$.

Proof. Let f be a checkable functions in NPSV/poly. We can suitably encode, for each x, the bits of $f(x)$ in a language A which is polynomial-time Turing equivalent to f and hence A is also checkable. The lowness result now follows by invoking Theorem 20. ■

Acknowledgements. The first author was partially supported by an Alexander von Humboldt fellowship and he is grateful to Prof. Uwe Schöning for hosting his visit to Ulm university where this work was carried out.

References

[1] V. Arvind and J. Köbler, *Pseudorandomness and resource-bounded measure*, Proceedings 17th Conference on the Foundations of Software Technology & Theoretical Computer Science, Springer-Verlag, LNCS 1346, pp. 235-249, 1997.

[2] V. Arvind, J. Köbler, and R. Schuler, *On helping and interactive proof systems*, International Journal of Foundations of Computer Science 6(2), 137-153, 1994.

[3] L. Babai, *Bounded round interactive proofs in finite groups*, SIAM Journal of Discrete Mathematics, 5: 88-111, 1992.

[4] N. Bshouty, R. Cleve, R. Gavaldà, S. Kannan, and C. Tamon, *Oracles and queries that are sufficient for exact learning*, Journal of Computer and System Sciences, 52, pp. 421–433 1996.

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

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

[7] J. L. Balcázar, J. Díaz, and J. Gabarró, *Structural Complexity I*, EATCS Monographs on Theoretical Computer Science. Springer-Verlag, second edition, 1995.

[8] M. Blum and S. Kannan, Designing programs that check their work, *Journal of the ACM*, **43**:269–291, 1995.

[9] J.Y. Cai, L.A. Hemaspaandra, and G. Wechsung, *Robust Reductions*, In *Proceedings of the 4th Annual International Computing and Combinatorics Conference*, Springer-Verlag, LNCS 1449, pp. 174-183, 1998.

[10] S. Fenner, L. Fortnow, A. Naik, and J. Rogers, *Inverting onto functions*, In Proceedings of the 11th IEEE Conference on Computational Complexity, IEEE, New York, pp. 213–222, 1996.

[11] O. Goldreich, S. Micali, and A. Wigderson, *Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems*, Journal of the ACM, 38:691–729, 1991.

[12] O. Goldreich and D. Zuckerman, *Another proof that BPP⊆PH (and more)*, Technical Report TR97-045, Electronic Colloquium on Computational Complexity, October 1997.

[13] 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.

[14] K. Ko and U. Schöning, *On circuit-size complexity and the low hierarchy in NP*, SIAM Journal on Computing, 14:41–51, 1985.

[15] J. Köbler, *On the structure of low sets*, In Proceedings of the 10th Structure in Complexity Theory Conference, 246–261. IEEE Computer Society Press, 1995.

[16] J. Köbler and U. Schöning, *High sets for NP*, In D. Zu and K. Ko, editors, Advances in Algorithms, Languages, and Complexity, pp. 139-156, Kluwer Acad. Publishers, 1997.

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

[18] J. Köbler and R. Schuler, *Average-case intractability vs. worst-case intractability*, Proceedings of the conference on Mathematical Foundations of Computer Sciences (MFCS)¡/I¿,¡BR¿ Springer-Verlag, LNCS 1450, 493-502, 1998. ¡BR¿ ¡P¿

[19] J. Köbler and O. Watanabe, *New collapse consequences of NP having small circuits*, Proceedings of the 22nd International Colloquium on Automata, Languages, and Programming, Springer-Verlag, LNCS 944, pp. 196–207, 1995.

[20] J. H. Lutz and W. J. Schmidt, *Circuit size relative to pseudorandom oracles*, Theoretical Computer Science, 107:95–120, 1993.

[21] N. Nisan and A. Wigderson, *Hardness vs randomness*, Journal of Computer and System Sciences, 49:149–167, 1994.

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

[23] M. Santha, *Relativized Arthur-Merlin versus Merlin-Arthur games*, Information and Computation, 80(1), pp. 44–49, 1989.

[24] U. Schöning, *A low and a high hierarchy within NP*, Journal of Computer and System Sciences, 27, pp. 14–28, 1983.

[25] U. Schöning, *Probabilistic complexity classes and lowness*, Journal of Computer and System Sciences, 39, pp. 84–100, 1989.

[26] A. Selman, *A taxanomy of complexity classes of functions*, Journal of Computer and System Sciences, 48(2), pp. 357–381, 1994.

ECCC ISSN 1433-8092
http://www.eccc.uni-trier.de/eccc
ftp://ftp.eccc.uni-trier.de/pub/eccc
ftpmail@ftp.eccc.uni-trier.de, subject 'help eccc'