*Electronic Colloquium on Computational Complexity* - Reports Series 1996 - available via:

ECCC
TR96-027

FTP: ftp.eccc.uni-trier.de:/pub/eccc/
WWW: http://www.eccc.uni-trier.de/eccc/
Email: ftpmail@ftp.eccc.uni-trier.de with subject 'help eccc'

# Computing Solutions Uniquely Collapses the Polynomial Hierarchy

Lane A. Hemaspaandra* Ashish V. Naik† Mitsunori Ogihara‡
Alan L. Selman§

**Abstract**

Is there an NP function that, when given a satisfiable formula as input, outputs one satisfying assignment uniquely? That is, can a nondeterministic function cull just one satisfying assignment from a possibly exponentially large collection of assignments? We show that if there is such a nondeterministic function, then the polynomial hierarchy collapses to $\mathrm{ZPP}^{\mathrm{NP}}$ (and thus, in particular, to $\mathrm{NP}^{\mathrm{NP}}$). As the existence of such a function is known to be equivalent to the statement "every NP function has an NP refinement with unique outputs," our result provides the strongest evidence yet that NP functions cannot be refined.

We prove our result via a result of independent interest. We say that a set $A$ is NPSV-selective (NPMV-selective) if there is a 2-ary partial NP function with unique values (a 2-ary partial NP function) that decides which of its inputs (if any) is "more likely" to belong to $A$; this is a nondeterministic analog of the recursion-theoretic notion of the semi-recursive sets and the extant complexity-theoretic notion of P-selectivity. Our hierarchy collapse result follows by combining the easy observation that every set in NP is NPMV-selective with the following result: If $A \in \mathrm{NP}$ is NPSV-selective, then $A \in (\mathrm{NP} \cap \mathrm{coNP})/\mathrm{poly}$. Relatedly, we prove that if $A \in \mathrm{NP}$ is NPSV-selective, then $A$ is $\mathrm{Low}_2$.

We prove that the polynomial hierarchy collapses even further, namely to NP, if all coNP sets are NPMV-selective. This follows from a more general result we prove: Every self-reducible NPMV-selective set is in NP.

*Dept. of Computer Science, University of Rochester, Rochester, NY 14627, USA. Supported in part by grants NSF-CCR-8957604, NSF-INT-9116781/JSPS-ENG-207, and NSF-CCR-9322513. Work done in part while visiting the University of Electro-Communications, Tokyo, Japan, and the Tokyo Institute of Technology.

†Dept. of Computer Science, SUNY–Buffalo, Buffalo, NY 14260, USA. Supported in part by grant NSF-CCR-9002292. Current affiliation: Department of Computer Science, University of Chicago, Chicago, IL 60637.

‡Dept. of Computer Science, University of Rochester, Rochester, NY 14627, USA. Supported in part by grants NSF-CCR-9002292 and NSF-INT-9116781/JSPS-ENG-207. Work done in part while visiting SUNY–Buffalo and while at the University of Electro-communications, Tokyo, Japan.

§Dept. of Computer Science, SUNY–Buffalo, Buffalo, NY 14260, USA. Supported in part by grants NSF-CCR-9002292, NSF-INT-9123551, and NSF-CCR-9400229.

# 1 Introduction

Valiant and Vazirani's [42] result that, in their words, "NP is as easy as detecting unique solutions," has rightly been the focus of great attention. Their breakthrough—a proof that every NP set *probabilistically* reduces to "detecting unique solutions" (technically, reduces to every solution to the promise problem (1SAT,SAT))—is one of the dual pillars on which Toda's [40] PH $\subseteq$ $\mathrm{P}^{\mathrm{PP}}$ paper rests, as do later papers extending Toda's result [41], and studying the complexity of function inversion [43,1].

Selman ([38], see also [12]) raised a related question that may be equally compelling, as he showed that a resolution would provide insight into the invertibility of honest polynomial-time functions, and into the relationship between single-valued and multivalued functions. He asked whether the following hypothesis is true.

**Hypothesis 1.1** *There is a* single-valued NP *function $f$ such that for each formula $F \in$ SAT, $f(F)$ is a satisfying assignment of $F$.*

Clearly, Hypothesis 1.1 is true if NP = coNP. However, as both Fenner et al. [12] and Selman [38] suspected that Hypothesis 1.1 fails, perhaps a more interesting issue is that of the evidential weight in that direction. In fact, little is currently known to indicate that Hypothesis 1.1 fails. The totality of current evidence seems to be the fact that Hypothesis 1.1 fails relative to a random oracle [33], and the result of Selman [38] that if Hypothesis 1.1 holds, then there are two disjoint NP-Turing-complete sets such that every set that separates them is NP-hard.

Since Hypothesis 1.1 is implied by NP = coNP, one might hope that Hypothesis 1.1 also implies a collapse of the polynomial hierarchy. The main result of this paper provides strong evidence that Hypothesis 1.1 fails: Hypothesis 1.1 implies that the polynomial hierarchy collapses to $\mathrm{ZPP}^{\mathrm{NP}}$ (and thus, in particular, to its second level, $\mathrm{NP}^{\mathrm{NP}}$). Equivalently, if all honest polynomial-time computable functions are NPSV-invertible, then the polynomial hierarchy collapses to $\mathrm{ZPP}^{\mathrm{NP}}$.

We obtain our result from a surprising and seemingly little-related direction: selectivity. Selectivity is a notion of generalized membership testing; selective sets have functions choosing which of any two input elements is the "more likely" to be in the set. Sets selective with respect to recursive selector functions were introduced by Jockush [20], and are called the *semirecursive* sets. Sets selective with respect to deterministic polynomial-time selector functions were introduced by Selman [36], and are called the P-selective sets; sets selective with respect to single-valued total NP functions were introduced and studied by Hemaspaandra et al. [19]. Recently, there has been a surge of interest in selective sets, and advances have catalyzed further advances (see the survey [9]).

In this paper, we extend the notion of selectivity, in the natural way, to functions that may be partial and/or multivalued. Important function classes of these sorts are the single-valued partial NP functions (NPSV), the multivalued partial NP functions (NPMV), and the multivalued total NP functions ($\mathrm{NPMV}_t$). Though it is easily observed that all NP sets are NPMV-selective, we will

prove the following result.

($\star\star$) If all NP sets are NPSV-selective then the polynomial hierarchy collapses to $\mathrm{ZPP}^{\mathrm{NP}}$.

It follows easily that Hypothesis 1.1 implies this same collapse.

Result ($\star\star$) is proven via the following result, which is of interest in its own right.

(1) The NPSV-selective sets in NP are in $(\mathrm{NP} \bigcap \mathrm{coNP})/\mathrm{poly}$.

$(\mathrm{NP} \bigcap \mathrm{coNP})/\mathrm{poly}$ is the class of sets (see [14]) accepted, aided by a small amount of "advice," by machines that robustly behave as $\mathrm{NP} \bigcap \mathrm{coNP}$ machines. We also prove the following related result.

(2) The NPSV-selective sets in NP are $\mathrm{Low}_2$.

That is, for each such set $A$, $\mathrm{NP}^{\mathrm{NP}^A} = \mathrm{NP}^{\mathrm{NP}}$. Though NPSV functions lack totality, the proofs of (1) and (2) show that one can nonetheless get the *effect* of totality in the cases that count—in particular, the definition of selectivity forces the functions to be defined whenever at least one input is in the fixed selective set. This will allow us to establish that the NPSV-selective sets in NP have lowness and advice class results just as strong as those shown by [19] for the $\mathrm{NPSV}_t$-selective sets in NP. The reason this advance is important is that results about $\mathrm{NPSV}_t$-selective sets offer no help in discrediting Hypothesis 1.1, but results about NPSV-selective sets do.

For coNP (and thus all higher levels of the polynomial hierarchy), an even stronger consequence can be obtained: All coNP sets are NPMV-selective if and only if NP = coNP. This result itself is a corollary of a more general result we prove: Every self-reducible NPMV-selective set is in NP. This contrasts with Buhrman, van Helden, and Torenvliet's result [8] that self-reducible P-selective sets are in P and with the result announced in [18] that self-reducible $\mathrm{NPMV}_t$-selective sets are in $\mathrm{NP} \cap \mathrm{coNP}$.

# 2 Definitions

Our alphabet will be $\Sigma = \{0, 1\}$. Let our pairing function $\langle \cdots \rangle$ be any "multi-arity onto," polynomial-time computable, polynomial-time invertible function (that is, the ranges of different arities are disjoint, and the union over all arities covers $\Sigma^*$, see, e.g., [16]).

For each partial, multivalued function $f$, *set-*$f(x)$ denotes the set of values of $f$ on input $x$. If $f(x)$ is undefined, then *set-*$f(x) = \emptyset$. We will use this notation for partial single-valued functions also, to avoid ambiguity regarding equality tests between potentially undefined values. For any two partial, multivalued functions $f$ and $g$, we say that $f$ is a *refinement* of $g$ if, for all $x$, it holds that

1. $f(x)$ is defined if and only if $g(x)$ is defined, and
2. *set-*$f(x) \subseteq$ *set-*$g(x)$.

We extend notions of selectivity [36,19] to multivalued and/or partial functions.

**Definition 2.1** *Let $\mathcal{FC}$ be any class of functions (possibly multivalued and/or partial). A set $A$ is $\mathcal{FC}$-selective if there is a function $f \in \mathcal{FC}$ such that for every $x$ and $y$, it holds that*

$$\textit{set-}f(x,y) \subseteq \{x,y\}, \textit{ and}$$
$$\textit{if } \{x,y\} \cap A \neq \emptyset, \textit{ then set-}f(x,y) \neq \emptyset \textit{ and set-}f(x,y) \subseteq A.$$

*By $\mathcal{FC}$-*sel *we denote the class of sets that are $\mathcal{FC}$-selective.*

We will be interested, in particular, in the following classes of functions.

**Definition 2.2 [6]**

1. NPMV *is the class of partial, multivalued functions $f$ for which there is a nondeterministic polynomial-time machine $N$ such that for every $x$, it holds that*
   (a) *$f(x)$ is defined if and only if $N(x)$ has at least one accepting computation path, and*
   (b) *for every $y$, $y \in$ set-$f(x)$ if and only if there is an accepting computation path of $N(x)$ that outputs $y$.*
2. NPMV$_t$ *is the class of total, multivalued functions in* NPMV.
3. NPSV *is the class of partial, single-valued functions in* NPMV.
4. NPSV$_t$ *is the class of total, single-valued functions in* NPMV.

Hypothesis 1.1 says that there is a partial function $f$ in NPSV such that for every formula $F$ in SAT, $f(F)$ is a satisfying assignment for $F$. It is trivial to observe that there is an NPMV function that finds *all* satisfying assignments of an input formula. Thus, the true complexity issue here is not of the complexity of finding satisfying assignments, but rather is of the complexity of *thinning down to one* the satisfying assignment set. Hypothesis 1.1 is equivalent to the assertion that all NPMV functions have refinements in NPSV ([38], see Proposition 3.1). We observe (Proposition 3.1) that Hypothesis 1.1 holds if and only if SAT is NPSV-selective.

Karp and Lipton introduced the following notion of being computable in a class supplemented by a small amount of extra information.

**Definition 2.3 [21]** *For any class of sets $\mathcal{C}$, $\mathcal{C}$/*poly *denotes the class of sets $L$ for which there exist a set $A \in \mathcal{C}$ and a polynomially length-bounded function $h : \Sigma^* \to \Sigma^*$ such that for every $x$, it holds that*

$$x \in L \textit{ if and only if } \langle x, h(0^{|x|}) \rangle \in A.$$

We will be particularly interested in the advice classes NP/poly, coNP/poly, and (NP $\bigcap$ coNP)/poly. It is not known whether NP/poly $\bigcap$ coNP/poly = (NP $\bigcap$ coNP)/poly, though Fenner et al. [11] have constructed an oracle relative to which the classes differ (see also the structural results of [14]).

Next we define lowness and extended lowness.

**Definition 2.4**

1. *[34] For each $k \geq 1$, define* $\mathrm{Low}_k = \{L \in \mathrm{NP} \mid \Sigma_k^{p,L} = \Sigma_k^p\}$, *where the $\Sigma_k^p$ are the $\Sigma$ levels of the polynomial hierarchy [30,39].*

2. *[27,4] For each $k \geq 2$, define* $\mathrm{ExtendedLow}_k = \{L \mid \Sigma_k^{p,L} = \Sigma_{k-1}^{p,\mathrm{SAT}\oplus L}\}$. *For each $k \geq 3$, define* $\mathrm{ExtendedLow}\Theta_k = \{L \mid \mathrm{P}^{(\Sigma_{k-1}^{p,L})[\mathcal{O}(\log n)]} \subseteq \mathrm{P}^{(\Sigma_{k-2}^{p,\mathrm{SAT}\oplus L})[\mathcal{O}(\log n)]}\}$. *The $[\mathcal{O}(\log n)]$ indicates that at most $\mathcal{O}(\log n)$ queries are made to the oracle.*

Hemaspaandra et al. [19] noted the following lowness and nonuniform class results for $\mathrm{NPSV}_t$-sel: $\mathrm{NPSV}_t\text{-sel} \subseteq (\mathrm{NP} \bigcap \mathrm{coNP})/\mathrm{poly}$, $\mathrm{NPSV}_t\text{-sel} \bigcap \mathrm{NP} \subseteq \mathrm{Low}_2$, and $\mathrm{NPSV}_t\text{-sel} \subseteq \mathrm{ExtendedLow}\Theta_3$.

Finally, we define "promise problems" [10] corresponding to selectivity. Informally, a solution to the promise problem PP-$A$ [37,28] will—if the promise is met that exactly one of $x$ and $y$ is in $A$—contain $\langle x, y\rangle$ exactly if $x \in A$.

**Definition 2.5 ([37], see also [28])** *Given any set $A$, we say that a set $B$* is a solution to PP-$A$ *if for every $\langle x, y\rangle$ such that exactly one of $x$ and $y$ is in $A$, $\langle x, y\rangle \in B$ if and only if $x \in A$.*

# 3 Unique Solutions Collapse The Polynomial Hierarchy

We first note a connection between refinements of NPMV functions, NPSV-selectivity, and inversion of polynomial-time functions. As is standard, we say a total polynomial-time computable function $f$ is *honest* if there is a polynomial $q$ such that, for all $x$, $q(|f(x)|) \geq |x|$. If $f$ is a (possibly non-1-to-1, possibly non-onto) total polynomial-time computable function, we say that $f$ is $\mathcal{C}$-invertible if there is a single-valued function $g$ in $\mathcal{C}$ such that $(\forall x)\,[(x \notin \mathrm{range}(f) \Rightarrow g(x) = \mathtt{undef})$ and $(x \in \mathrm{range}(f) \Rightarrow f(g(x)) = x)]$ (see [2,15,43,38] for a detailed discussion of invertibility). Observe that $f$ is $\mathcal{C}$-invertible if and only if the partial multivalued function $f^{-1}$ has a single-valued refinement in $\mathcal{C}$. NP2V is the class of all NPMV functions $f$ such that $(\forall x)\,[\,||\, \textit{set-}f(x) \,||\, \leq 2]$.

**Proposition 3.1** *(see also [38]) The following are equivalent:*

1. *Hypothesis 1.1 holds.*
2. *All* NPMV *functions have* NPSV *refinements.*
3. *All* NP2V *functions have* NPSV *refinements.*
4. SAT *is* NPSV-*selective.*
5. *All* NP *sets are* NPSV-*selective.*
6. *All honest* FP *functions are* NPSV-*invertible.*

**Proof of Proposition 3.1** The reader may easily observe that every set in NP is NP2V-selective. Note also that any NPSV refinement of an NPMV-selector for a set is itself an NPSV-selector for the set. Thus, Part 3 implies Part 5. Clearly, Part 5 implies Part 4, and Part 2 implies Part 3. Part 4 implies Part 1, as an NPSV function $f'$ that is an NPSV-selector for SAT could be used to create the function $f$ from the statement of Hypothesis 1.1 as follows. Let $f$ be the function that on an input formula $F$ simulates $f'$ applied to the top node of $F$'s 2-disjunctive-self-reduction tree, and then each path of (the simulation of) $f$ that gets an output simulates $f$ applied to 2-disjunctive-self-reduction of the output node, and so on, and that at any reached leaf of the self-reduction tree checks that the leaf is a satisfying assignment and outputs it if it is. Finally, Selman [38] has noted that Parts 1, 2, and 6 are equivalent, Part 6 being equivalent by combining [38, Exercise 5] with the comment in the last paragraph of [38, Section 1.2]. ∎

Naik, Regan, Royer, and Selman (in preparation) have noted that this behavior applies not just to the classes mentioned here, but to any class having certain nice closure properties.

Our hierarchy result will follow easily from our study of the lowness and circuit properties of the new selectivity classes we've mentioned—the NPSV-selective sets, the NPMV-selective sets, and the NPMV$_t$-selective sets. We now turn to this study, emphasizing the NPSV-selective sets. Clearly, NPMV-sel (and thus NPSV-sel) is contained in NP/poly, and NPMV$_t$-sel is contained in NP/poly $\bigcap$ coNP/poly, via using a standard divide-and-conquer approach to find an appropriate advice set, similar to the approach in Ko's proof [22] that the P-selective sets are in P/poly (see also the proofs of Theorem 3.2 and Theorem 3.7). We conclude, via the extended lowness of NP/poly $\bigcap$ coNP/poly (Theorem 3.4) and the lowness of NP/poly $\bigcap$ coNP/poly $\bigcap$ NP = coNP/poly $\bigcap$ NP [44], that NPMV$_t$-sel is ExtendedLow$_3$ and that NPMV$_t$-sel $\bigcap$ NP is Low$_3$. We now turn towards our main result.

**Theorem 3.2** NPSV-sel $\bigcap$ NP $\subseteq$ (NP $\bigcap$ coNP)/poly.

In the introduction, we mentioned that the key thing our proofs do is to achieve, even with the partial functions, the *effect* of totality. In the proof of Theorem 3.2, it is easy to put one's finger on the exact part of the construction that achieves this—our decision to require the advice string to encode certificates. This decision allows what would otherwise be an NP/poly $\bigcap$ coNP/poly containment to become an (NP $\bigcap$ coNP)/poly containment, as the fact that the advice contains certificates allows an NP $\bigcap$ coNP machine to verify whether or not the strings purported to be from the set in fact are from the set, and this itself allows the machine to be robustly NP $\bigcap$ coNP-like, that is, NP $\bigcap$ coNP-like for all possible advice strings, even incorrect ones.

**Proof of Theorem 3.2** Let $A \in$ NP be NPSV-selective, with selector function $f \in$ NPSV. Without loss of generality we assume $f$ satisfies $(\forall x, y)[set\text{-}f(x,y) = set\text{-}f(y,x)]$, since if it doesn't, we can replace it with $f\text{-}new(a,b) = f(\min(a,b), \max(a,b))$. It is trivial to create an appropriate advice string at lengths $n$ for which $||A^{\leq n}|| = 0$, so we assume this is done tacitly at such lengths, and below consider just the $||A^{\leq n}|| \neq 0$ case. Recall that $set\text{-}f(x) = \{y \mid y \text{ is a value of } f(x)\}$. Let $p$ be a monotone nondecreasing polynomial and $B$ be a set in P witnessing that $A \in$ NP so that for

every $x$, $x \in A$ if and only if for some $y \in \Sigma^{p(|x|)}$, $\langle x, y\rangle \in B$. Let $w$ be a string of the form $\langle 0^n, S, T\rangle$, where $S$ and $T$ encode finite sets. We call $w$ an *advice string* for $n$ if (i) $||\,T\,|| \leq ||\,S\,|| \leq n+1$, (ii) $S \subseteq \Sigma^{\leq n}$, (iii) $T \subseteq \Sigma^{\leq p(n)}$, and (iv) for every $y \in S$, there is some $z \in T$ such that $\langle y, z\rangle \in B$, that is, $y \in A$ is certified by $z$. Moreover, $w$ is called a *good advice string* for $n$ if it holds that

**(*)** $(\forall x \in \Sigma^{\leq n})[x \in A \Rightarrow (\exists y \in S)[\mathit{set\text{-}f}(x, y) = \{x\}]]$.

For every $n$, a good advice string for $n$ exists. As in the case of Ko's proof that the P-selective sets are in P/poly [22], we may repeatedly choose to add to $S$ some element of $A^{\leq n}$ that loses to at least half the elements that both are not yet in $S$ and don't yet beat some element in $S$, where by "$x$ loses to $y$" we mean that $\mathit{set\text{-}f}(x, y) = \{y\}$. Since $||\,\Sigma^{\leq n}\,|| < 2^{n+1}$, $S$ will have at most $n+1$ elements. After constructing $S$, for each $y \in S$, we pick up one certificate and construct $T$.

Clearly, the set of all advice strings is in P. Moreover, the set of all good advice strings is in coNP. As $w = \langle 0^n, S, T\rangle$ being an advice string guarantees that $S \subseteq A$, $\mathit{set\text{-}f}(x, y)$ is defined for any $x \in \Sigma^{\leq n}$ and $y \in S$. So, $w = \langle 0^n, S, T\rangle$ is a good advice string for $n$ if and only if

$$w \text{ is an advice string and } (\forall x \in \Sigma^{\leq n})[x \notin A \vee (\exists y \in S)[y = x \ \vee \ y \notin \mathit{set\text{-}f}(x, y)]].$$

Clearly, this is a coNP-predicate as, in particular, testing $y \notin \mathit{set\text{-}f}(x, y)$ can be done via one universal quantification. However, note that if $w$ is an advice string for $n$, then for every $x \in \overline{A}^{\leq n}$ and $y \in S$, $\mathit{set\text{-}f}(x, y) = \{y\} \neq \{x\}$. So, if $w = \langle 0^n, S, T\rangle$ is a good advice string for $n$, then

**($\star$)** $(\forall x \in \Sigma^{\leq n})[x \in A \iff (\exists y \in S)[\mathit{set\text{-}f}(x, y) = \{x\}]]$.

Now define

$$\begin{aligned} A' \quad = \quad & \{\langle x, \langle 0^{|x|}, S, T\rangle\rangle \mid \langle 0^{|x|}, S, T\rangle \text{ is an advice string for } |x|, \text{ and} \\ & (\exists y \in S)[\mathit{set\text{-}f}(x, y) = \{x\}]\}. \end{aligned}$$

Note that $A' \in \text{NP} \bigcap \text{coNP}$. The containment in NP is immediate. The containment in coNP follows from the fact that, as long as $\langle 0^{|x|}, S, T\rangle$ is an advice string for $|x|$, $S \subseteq A$ guarantees that $\mathit{set\text{-}f}(x, y)$ is either $\{x\}$ or $\{y\}$ for any $y \in S$.

Now for each $n$, define $h(0^n)$ to be the smallest good advice string for $n$ in lexicographic order. Then, by ($\star$), for every $x$, $x \in A$ if and only if $\langle x, h(0^{|x|})\rangle \in A'$. This proves that $A \in (\text{NP} \bigcap \text{coNP})/\text{poly}$. ∎

Theorem 3.3 follows from essentially the same proof as that of Theorem 3.2.

**Theorem 3.3** NPSV-sel $\subseteq$ NP/poly $\bigcap$ coNP/poly.

This result reflects a more general behavior. By $\mathit{graph}(f)$, we denote $\{\langle x, y\rangle \,\big|\, y \in \mathit{set\text{-}f}(x)\}$. Let $\mathcal{FC}$ be any function class (possibly partial, possibly multivalued). Let $\mathcal{C}$ be any class having the property that for each $f$ in $\mathcal{FC}$ it holds that $\mathit{graph}(f) \in \mathcal{C}$. Then

$$\mathcal{FC}\text{-sel} \subseteq (R^p_{dtt}(\mathcal{C}))/\text{poly}.$$

In particular, the polynomial advice represents advice strings found by divide and conquer, and the disjunctive queries determine, via the graph of the selector function, the action of the selector function on the input paired with each string in the advice set, and additionally the disjunctive reducer checks whether the input is one of the advice strings. The reducer accepts exactly when the input either is one of the advice strings or is an output of the selector function when that function is run on the input paired with one of the advice strings (see the proofs of Theorem 3.2 above and Theorem 3.7 below). Theorem 3.3 is a specific case of this more general claim. The polynomial time-bound on the disjunctive reduction in the general claim can be replaced by a logspace bound if the pairing function used (in the definition of advice classes) is logspace invertible.

Köbler [23] has shown that (NP $\bigcap$ coNP)/poly is ExtendedLow$\Theta_3$. An interesting question left open by Köbler's paper is whether (NP/poly) $\bigcap$ (coNP/poly) is extended low. We resolve this issue by showing that it is. It is an interesting open issue whether our result can itself be strengthened via the techniques of Gavaldà and Köbler [13,23] to an ExtendedLow$\Theta_3$ result; we conjecture that it can. In any case, in terms of the standard levels of extended lowness—ExtendedLow$_1$, ExtendedLow$_2$, ExtendedLow$_3$, ...—our ExtendedLow$_3$ result *is* optimal, as Allender and Hemaspaandra [3] have noted that even P/poly is not in ExtendedLow$_2$. We defer the proof of Theorem 3.4 to the end of this section.

**Theorem 3.4** (NP/poly) $\bigcap$ (coNP/poly) *is* ExtendedLow$_3$.

From Theorems 3.3 and 3.4, we immediately obtain the following corollary.

**Corollary 3.5** *The* NPSV-*selective sets are* ExtendedLow$_3$.

What can be said about the lowness of the NPSV-selective sets in NP? Theorem 3.3 and Köbler's "(NP $\bigcap$ coNP)/poly $\bigcap$ NP is Low$\Theta_3$" result imply a Low$\Theta_3$ result. However, as the next corollary states, the NPSV-selective sets in NP are in fact Low$_2$. Informally, the reason for this improvement is that NPSV-selective sets have selector functions that, while perhaps partial, are sharply constrained. In particular, these functions are only partially partial. They are forced to be total whenever either of the inputs is in the given set, and, as we did also in the proof of Theorem 3.2, we exploit this conditional totality in our Low$_2$ proof below.

**Lemma 3.6 [28]** *If $A$ is in $\Sigma^p_i$ and $B$ is a solution of* PP-$A$, *then $\Sigma^{p,A}_{i+1} \subseteq \Sigma^{p,B}_{i+1}$.*

**Theorem 3.7** *If $A \in$* NPSV-sel $\bigcap$ NP, *then* PP-$A$ *has a solution $L$ that is* Low$_2$.

Corollary 3.8 follows immediately from Theorem 3.7 via Lemma 3.6.

**Corollary 3.8** NPSV-sel $\bigcap$ NP $\subseteq$ Low$_2$.

**Proof of Theorem 3.7** Let $A \in$ NPSV-sel $\cap$ NP, with selector function $f \in$ NPSV. As in the proof of Theorem 3.2, define the notion of advice strings and good advice strings. Define

$$\widehat{A} = \{\langle x, y\rangle \mid \textit{set-}f(x, y) = \{x\} \text{ and } x \in A\}.$$

Clearly, $\widehat{A}$ is a solution of PP-$A$ and is in NP. It suffices to show that $\Sigma_2^{p,\widehat{A}} \subseteq \Sigma_2^p$. Let $w = \langle 0^n, S, T\rangle$ be a good advice string for $n$. Then for every $x \in \Sigma^{\leq n}$,

$$x \in A \iff (\exists y \in S)[\mathit{set\text{-}f}(x, y) = \{x\}].$$

So, for every $x, y \in \Sigma^{\leq n}$,

$$\begin{aligned}
\langle x, y\rangle \notin \widehat{A} &\iff x \notin A \vee \mathit{set\text{-}f}(x, y) \neq \{x\} \\
&\iff x \notin A \vee (x \in A \wedge \mathit{set\text{-}f}(x, y) \neq \{x\}) \\
&\iff (\forall z \in S)[\mathit{set\text{-}f}(x, z) \neq \{x\}] \vee (x \in A \wedge \mathit{set\text{-}f}(x, y) \neq \{x\}) \\
&\iff (\forall z \in S)[\mathit{set\text{-}f}(x, z) = \{z\}] \vee (x \in A \wedge \mathit{set\text{-}f}(x, y) = \{y\}).
\end{aligned}$$

Define $T = \{\langle x, y, \langle 0^n, S, T\rangle\rangle \mid |x|, |y| \leq n, w = \langle 0^n, S, T\rangle$ is an advice string for $n$, and either $(\forall z \in S)[\mathit{set\text{-}f}(x, z) = \{z\}]$ or $x \in A \wedge \mathit{set\text{-}f}(x, y) = \{y\}\}$. Then, $T \in$ NP, and for every good advice string $w = \langle 0^n, S, T\rangle$ and $x, y$ of length at most $n$, $\langle x, y\rangle \notin \widehat{A}$ if and only if $\langle x, y, w\rangle \in T$.

Now let $C \in \Sigma_2^{p,\widehat{A}}$ and let $N_1$ and $N_2$ be NP-machines such that $C = L(N_1, L(N_2, \widehat{A}))$. There is a polynomial $q$ such that for every $x$ and every possible query $y$ of $N_1$ on $x$, if $N_2$ on $y$ queries $\langle u, v\rangle$, then $|u|, |v| \leq q(|x|)$. Define $D$ to be the set of all $\langle y, \langle 0^m, S, T\rangle\rangle$ such that

- $w = \langle 0^m, S, T\rangle$ is an advice string for $m$ and
- there is an accepting computation path $\pi$ of $N_2$ on $y$ such that for every query $\langle u, v\rangle$ along path $\pi$,
  - $|u|, |v| \leq m$,
  - if the answer to the query is affirmative, then $\langle u, v\rangle \in \widehat{A}$, and
  - if the answer to the query is negative, then $\langle u, v, w\rangle \in T$.

Since both $\widehat{A}$ and $T$ are in NP, $D \in$ NP. Furthermore, if $y$ is a query of $N_1$ on $x$, then for every good advice string $w$ for $q(|x|)$, $N_2^{\widehat{A}}$ on $y$ accepts if and only if $\langle y, w\rangle \in D$.

Now define $E$ to be the set of all $\langle x, w\rangle$ such that $w$ is an advice string for $q(|x|)$ and $N_1$ on $x$ accepts if its query $y$ is answered affirmatively if and only if $y \in D$. Since $D$ is in NP, $E \in \Sigma_2^p$. Furthermore, for every $x$ and good advice string $w$ for $q(|x|)$, $\langle x, w\rangle \in E$ if and only if $x \in C$. Therefore, for every $x$, $x \in C$ if and only if there is a good advice string $w$ for $q(|x|)$ such that $\langle x, w\rangle \in E$. As described in the proof of Theorem 3.2, the set of all good advice strings is in coNP. Thus, $C \in \Sigma_2^p$. This proves the theorem. ∎

Note that every NP set is NPMV-selective. Is this also true for NPSV-selectivity? We have the following result.

**Theorem 3.9** *If* NP $\subseteq$ NPSV-sel, *then* $\mathrm{ZPP}^{\mathrm{NP}} = \mathrm{PH}$.

**Proof of Theorem 3.9** This is a corollary of Theorem 3.2, since, extending Karp and Lipton [21], Köbler and Watanabe have proven that if NP $\subseteq$ (NP $\bigcap$ coNP)/poly $\Rightarrow$ ZPP$^{\mathrm{NP}}$ = PH [24]. ∎

Note that we could conclude immediately from Corollary 3.8 the slightly weaker result that if NP $\subseteq$ NPSV-sel, then NP$^{\mathrm{NP}}$ = PH.

From Proposition 3.1 and Theorem 3.9, we have our main result, and a related result.

**Corollary 3.10** *If Hypothesis 1.1 is true then* ZPP$^{\mathrm{NP}}$ = PH.

**Corollary 3.11** *If all honest* FP *functions are* NPSV-*invertible then* ZPP$^{\mathrm{NP}}$ = PH.

Hypothesis 1.1 seems somewhat akin to the statement UP=NP, in the sense that both speak of reducing a multiplicity (respectively of values and of certificates) to a unity. However, NP might be equal to UP because of the existence of some strange machine that accepts SAT uniquely and has nothing to do with finding satisfying assignments, and, on the other hand, there might exist a machine that outputs satisfying assignments uniquely but "ambiguously"—along more than one computation path. Indeed, it remains an open question whether either of UP=NP and Hypothesis 1.1 implies the other. It also remains an open question whether Corollary 3.10 remains true if the hypothesis is changed to UP=NP; indeed, it is not even known whether UP=NP implies that the polynomial hierarchy collapses at any level. It is easily seen, as noted by Buhrman, Kadin, and Thierauf [7], that SAT has an NPSV refinement if and only if it has (in a certain model for oracle access to partial functions) an FP$^{\mathrm{NPSV[1]}}$ refinement, and thus Corollary 3.10 speaks to that case.

We conclude this section with the deferred proof of Theorem 3.4.

**Proof of Theorem 3.4** Let $H \in (\mathrm{NP/poly}) \bigcap (\mathrm{coNP/poly})$. Let $B \in \mathrm{NP}^{\mathrm{NP}^{\mathrm{NP}^H}}$ (let's say, for convenience, $B = L(N_1^{L(N_2^{L(N_3^H)})})$). We will show that $B \in \mathrm{NP}^{\mathrm{NP}^{\mathrm{SAT}\oplus H}}$.

Let $S_1$ ($S_2$) be an NP (coNP) set certifying $H \in \mathrm{NP/poly}$ ($H \in \mathrm{coNP/poly}$). Let $p(\cdot)$ be a polynomial bounding the size of the correct advice sequences for each. Let $q(\cdot)$ be a polynomial composing the polynomial running times of $N_1$, $N_2$, and $N_3$.

Recall that our pairing function, $\langle \cdots \rangle$, is some nice, "multi-arity onto" pairing function. On input $x$, our base NP machine of our $\mathrm{NP}^{\mathrm{NP}^{\mathrm{SAT}\oplus H}}$ machine guesses nondeterministically strings $r_1$, …, $r_{q(|x|)}$, and $s_1$, …, $s_{q(|x|)}$, satisfying, for each $i$, $|r_i| \leq p(i)$ and $|s_i| \leq p(i)$. Via a single call to $\mathrm{NP}^{\mathrm{SAT}\oplus H}$, the base machine checks whether $r_1$, …, $r_{q(|x|)}$ is a good advice set for helping $S_1$. In particular, we make one query, $\langle x, r_1, \ldots, r_{q(|x|)} \rangle$, to the $\mathrm{NP}^{\mathrm{SAT}\oplus H}$ set:

$$E' = \{ \langle x, r_1, \cdots, r_z \rangle \mid z = q(|x|) \text{ and}$$

$$(\forall i : 1 \leq i \leq z)\,[|r_i| \leq p(i)] \text{ and } (\exists y : |y| \leq q(|x|))\,[y \in H \iff \langle y, r_{|y|} \rangle \notin S_1] \},$$

and if the answer is "no," we know the "$r$" advice collection is good. Similarly, with one question to an $\mathrm{NP}^{\mathrm{SAT}\oplus H}$ set $E''$ (defined analogously), we determine whether the "$s$" advice collection is good for helping $S_2$.

Note that *when given the correct advice strings*, an NP machine can strongly (in the sense of Long [26] and Selman [35]) check whether $x \in H$ or $x \notin H$, by nondeterministically guessing which is true and checking an $x \in H$ guess via checking whether $\langle x, r_{|x|} \rangle \in S_1$, and checking an $x \notin H$ guess via checking whether $\langle x, s_{|x|} \rangle \in S_2$.

Our simulation of $B = L(N_1^{L(N_2^{L(N_3^H)})})$ in $\mathrm{NP}^{\mathrm{NP}^{\mathrm{SAT} \oplus H}}$ proceeds as follows (for simplicity, let's call our base machine $N_4$). $N_4$ guesses and checks good advice sets as already described. $N_4$ now simulates $N_1$, except each time $N_1$ asks a query $y$ to $L(N_2^{L(N_3^H)})$, $N_4$ asks the query $\langle y, \langle r_1, \ldots, r_{q(|x|)} \rangle, \langle s_1, \ldots, s_{q(|x|)} \rangle \rangle$ to an $\mathrm{NP}^{\mathrm{SAT} \oplus H}$ set $E'''$, which itself will satisfy $E''' = L(N_5^{\mathrm{SAT} \oplus H})$ for a machine $N_5$ to be defined. (Since we have only one $\mathrm{NP}^{\mathrm{SAT} \oplus H}$ oracle, the actual set we will use is $E = E' \oplus E'' \oplus E'''$.) $N_5$ on input $\langle y, \langle r_1, \ldots, r_t \rangle, \langle s_1, \ldots, s_t \rangle \rangle$ simulates $N_2$ on input $y$, except every time $N_2$ asks a query $z$ to $L(N_3)$ on input $y$, $N_5$ asks the query $\langle z, \langle r_1, \ldots, r_t \rangle, \langle s_1, \ldots, s_t \rangle \rangle$ to the NP set $G$ (since SAT is NP-complete, we implicitly convert the query to an appropriate query to SAT):

> $G = \{ \langle z, \langle r_1, \ldots, r_t \rangle, \langle s_1, \ldots, s_t \rangle \rangle \mid$ if we simulate $N_3^H(z)$, replacing each call to $H$ (say "$w \in H$?") by nondeterministically checking whether $\langle w, r_{|w|} \rangle \in S_1$ (in which case we proceed along the path certifying $\langle w, r_{|x|} \rangle$ as of $w \in H$) and (separately, nondeterministically) whether $\langle w, s_{|w|} \rangle \notin S_2$ (in which case we proceed as if $w \notin H$), we have an accepting path of our simulated $N_3 \}$. Note: if any of the $w$ are such that $|w| > t$, we act as if $s_{|w|} = r_{|w|} = \epsilon$, as in actual runs this case will not occur.

We make no claim that $G \in \mathrm{NP} \bigcap \mathrm{coNP}$. In fact, with "bad" advice as inputs, the simulation defining $G$ will be quite chaotic: a query "$w \in H$?" might be treated as being answered both "yes" and "no," or neither "yes" nor "no." However, *when given good advice sets*, the machine will in fact correctly simulate $N_3^H(z)$: each query $w$ of $N_3(z)$ will be answered either "yes" or "no," will not be answered both "yes" and "no," and will be answered correctly. That is, $G$'s simulation of $H$ is, *when the advice is correct*, an example of strong computation. Crucially, for every query actually asked of $G$ during an actual run of our $\mathrm{NP}^{E' \oplus E'' \oplus L(N_5^G)}$ algorithm, the advice will be correct (and thus the strong computation going on within $G$ will be correct). Recall that this behavior, in which every *actual access* to an oracle maintains a certain nice property of the oracle computation (such as computing strongly), though some queries that are never asked might taint the property, is known as "guarded" access. We've now given an $\mathrm{NP}^{\mathrm{NP}^{\mathrm{SAT} \oplus H}}$ simulation of an arbitrary set $B \in \mathrm{NP}^{\mathrm{NP}^{\mathrm{NP}^H}}$, for arbitrary $H \in (\mathrm{NP/poly}) \bigcap (\mathrm{coNP/poly})$. ∎

# 4 NPMV-Selectivity versus Self-reducibility

Buhrman, van Helden, and Torenvliet [8] showed that if a self-reducible set is P-selective, then it is in P, and Hemaspaandra et al. [18] proved that if a self-reducible set is $\mathrm{NPMV}_t$-selective, then it

is in NP $\cap$ coNP. We prove, as Theorem 4.3 below, a similar result for self-reducible NPMV-selective sets, and apply this result to PSPACE and the levels of the polynomial hierarchy.

The standard definition of self-reducibility that is used in most contemporary research in complexity theory was given by Meyer and Paterson [29].

**Definition 4.1** *A polynomial time computable partial order $<$ on $\Sigma^*$ is* OK *if and only if*

1. *each strictly decreasing chain is finite and there is a polynomial $p$ such that every finite $<$-decreasing chain is shorter than $p$ of the length of its maximum element, and*
2. *for all $x, y \in \Sigma^*$, $x < y$ implies that $|x| \leq p(|y|)$.*

**Definition 4.2** *A set $L$ is* self-reducible *if there is an OK partial order $<$ and a deterministic polynomial time-bounded machine $M$ such that $M$ accepts $L$ with oracle $L$ and, on any input $x$, $M$ asks its oracle only about words strictly less than $x$ in the OK partial order $<$. If the self-reduction of the query machine $M$ in fact is also a polynomial-time disjunctive (conjunctive) truth-table reduction, then $L$ is disjunctive (conjunctive) self-reducible.*

Note in particular that unless otherwise specified we use self-reducible to mean Turing self-reducible.

**Theorem 4.3** *If $A$ is self-reducible and* NPMV*-selective, then $A \in$ NP.*

**Proof of Theorem 4.3** First, we need to introduce some notation. Let $B$ be a set and $S$ be a finite set. Let $\succeq$ be a total order over $S$ such that for every $x, y \in S$, $x \succeq y \iff (x \in B \Rightarrow y \in B)$. Then for each $x, y \in S$, define $x \equiv y$ if there exist some $w_1, \cdots, w_m \in S$ such that (i) both $x$ and $y$ appear in $w_1, \cdots, w_m$, (ii) $w_m = w_1$, and (iii) for every $i, 1 \leq i \leq m-1$, $w_i \succeq w_{i+1}$, and define $x \succ y$ if $x \succeq y$ and $x \not\equiv y$. Call $x \in S$ *minimal* if for every $y \in S$, either $x \equiv y$ or $x \succ y$. Note that $x \equiv y$ implies $x \in B$ if and only if $y \in B$, and therefore, for any minimal $x$, $x \in B$ implies $S \subseteq B$. Also note that finding all minimal elements in $S$ is equivalent to dividing a "directed clique" (i.e., a clique in which each edge is directed) into its fully connected components and finding the (necessarily unique) component from which no other component is reachable. So, after knowing whether $x \succeq y$ or $y \succeq x$ for each $x, y \in S$, finding all minimal elements can be done in time polynomial in $\sum_{x \in S} |x|$.

Let $A$ be self-reducible via a machine $M$ and an OK partial order $<$ as in Definition 4.2. Let $\perp$ be a fixed element in $A$ and, without loss of generality, assume for every $x \in \Sigma^*$ other than $\perp$ that $\perp < x$. Let $A$ be NPMV-selective with selector function $f \in$ NPMV. Consider the nondeterministic Turing machine $N$ defined, on input $x$, by the following algorithm.

**(1)** Nondeterministically guess one computation path of $M$ on $x$ together with oracle answers and put into $S_1$ ($S_0$) all the queries for which affirmative (negative) answers are guessed.

If $M$ on $x$ along the guessed path rejects, then reject $x$.

**(2)** For each $y \in S_0$ and $z \in S_1 \cup \{x\}$, nondeterministically verify that $z \in$ *set-$f$*$(y, z)$.

If the verification is not successful for some $y, z$, then reject $x$.

**(3)** For each $y, z \in S_1$, nondeterministically compute $f(y,z)$ and define $y \succeq z$ if $f(y,z) = z$ and $z \succeq y$ if $f(y,z) = y$.

If for some $y$ and $z$, computing $f(y,z)$ is not successful, then reject $x$.

**(4)** If $S_1 = \emptyset$, then output $\perp$. Otherwise, output lexicographically the smallest minimal string in $S_1$.

It is easy to see that $N$ is polynomial-time bounded. We claim the following:

- For every $x \notin A$, if $N$ on $x$ outputs $y$, then $y < x$ and $y \notin A$.
- For every $x \in A$,
  - $N$ on $x$ has an output in $A$ and
  - every output $y$ of $N$ on $x$ satisfies $(y = \perp) \vee (y < x)$.

This is seen as follows. Suppose that $x \notin A$ and $N$ on $x$ outputs $w$ at step (4). As $N$ must choose an accepting computation path of $M$ on $x$, either $S_0 \not\subseteq \overline{A}$ or $S_1 \not\subseteq A$. But, the former is not the case, for, since the verifications in step (2) are all successful, $S_0$ having an element in $A$ implies $x \in A$, yielding a contradiction. So, the latter is the case. Since $w$ is minimal in $S_1$, $w \in A$ implies $S_1 \subseteq A$. So, $w$ cannot be in $A$. Hence the first claim holds.

On the other hand, suppose that $x \in A$. The machine $N$ can guess the "correct" accepting computation path of $M$ on $x$, for which $S_0 \subseteq \overline{A}$ and $S_1 \subseteq A$. After guessing the path, $N$ can reach step (4) because for every $y \in S_0$ and $z \in S_1$, $\mathit{set}\text{-}f(y,z) = \{z\}$, and for every $y, z \in S_1$, $\mathit{set}\text{-}f(y,z) \neq \emptyset$. After entering step (4), $N$ will choose its output from $\{\perp\} \cup S_1$, which is a subset of $A$. So, $N$ will output a string in $A$. Hence the second claim holds.

Now consider a machine $D$ that, on input $x$, starting with $w = x$, executes the following algorithm.

**(I)** Simulate $N$ on $w$.

**(II)** If $N$ rejects, then reject. If $N$ outputs $\perp$, then accept. Otherwise, set $w$ to the output of $N$ and go back to (I)

By the definition of self-reducibility, step (I) is repeated at most polynomially many times, and thus, $D$ is polynomial-time bounded. By the above two claims, if $x \notin A$, then $D$ never obtains $\perp$ as the output of $N$, and if $x \in A$, for some computation path, $D$ obtains $\perp$ as the output of $N$. So, $D$ accepts $x$ if and only if $x \in A$. This establishes that $A \in$ NP. ∎

Note that from the well-known fact that every disjunctive self-reducible set is in NP and from the fact that every set in NP is NPMV-selective, it follows that every disjunctive self-reducible set is NPMV-selective. Theorem 4.3 yields, for example, the following consequences, keeping in mind

the fact that PSPACE and each $\Sigma_k^p$ have self-reducible complete sets. Note that the $k \geq 2$ below cannot be improved to $k \geq 1$ unless PH = NP.

**Corollary 4.4**

1. PSPACE $\subseteq$ NPSV-sel *if and only if* PSPACE $\subseteq$ NPMV-sel *if and only if* PSPACE = NP.
2. *For any* $k \geq 2$, $\Sigma_k^p \subseteq$ NPSV-sel *if and only if* $\Sigma_k^p \subseteq$ NPMV-sel *if and only if* PH = NP.
3. coNP $\subseteq$ NPMV-sel *if and only if* NP = coNP.

# 5 Conclusion and Open Questions

This paper has studied the complexity of computing a *single* satisfying assignment of an input satisfiable formula. Previously, it was (trivially) known that satisfying assignments could be found by polynomial-time functions if and only if P=NP. It was also (trivially) known that an assignment could be found via a polynomial-time machine using an NP oracle (and it was known, not trivially, that finding the lexicographically largest assignment was the hardest of all problems solvable in that class [25]).

But what about function classes between FP and $\mathrm{FP}^{\mathrm{NP}}$? In this paper, we proved that the NPSV functions are unlikely to have the power to find satisfying assignments; they can do so only if the polynomial hierarchy collapses to $\mathrm{ZPP}^{\mathrm{NP}}$. There remains an important function class intermediate in power between the NPSV functions (shown by this paper to be unlikely to have the power of finding satisfying assignments) and the functions computable via Turing access to an NP oracle (which clearly can find satisfying assignments). This class is the class of (partial) functions computable via *parallel* (that is, truth-table) access to an NP oracle. Clearly, NPSV is a subset of this class (cf. [38]). The key open issue is distilled in the following hypothesis (see [43,1,17,31,38] for background and discussion).

**Hypothesis 5.1** *Every* NPMV *function has a (single-valued) refinement in* $\mathrm{FP}_{tt}^{\mathrm{NP}}$.

The above can be equivalently phrased as: There is a partial function $f$ computable by a polynomial-time Turing machine making parallel queries to NP such that for each formula $F \in$ SAT, $f(F)$ is a satisfying assignment of $F$. Does Hypothesis 5.1 imply a collapse of the polynomial hierarchy? It seems that such a result would require techniques substantially different from those of this paper. In particular, our result that "NPMV has NPSV refinements only if $\mathrm{ZPP}^{\mathrm{NP}} = \mathrm{PH}$" itself relativizes. However, any relativizable proof of "Hypothesis 5.1 implies a collapse of the polynomial hierarchy" would immediately imply—due to the result of Watanabe and Toda [43] that Hypothesis 5.1 holds relative to a random oracle—that the polynomial hierarchy collapses relative to a random oracle. Furthermore, if the polynomial hierarchy collapses relative to a random oracle, then the polynomial hierarchy collapses ([5], see also [32]). The main result of the present paper does not similarly imply

that the polynomial hierarchy collapses relative to a random oracle, as Hypothesis 1.1 is known to fail relative to a random oracle [33].

**Acknowledgments**

The authors would like to thank S. Biswas, H. Buhrman, L. Fortnow, Y. Han, E. Hemaspaandra, and M. Zimand for many helpful comments and suggestions.

# References

[1] K. Abrahamson, M. Fellows, and C. Wilson, *Parallel self-reducibility*, in Proceedings of the 4th International Conference on Computing and Information, IEEE Computer Society Press, May 1992, pp. 67–70.

[2] E. Allender, *Invertible functions*, 1985. PhD thesis, Georgia Institute of Technology.

[3] E. Allender and L. Hemachandra, *Lower bounds for the low hierarchy*, Journal of the ACM, 39 (1992), pp. 234–251.

[4] J. Balcázar, R. Book, and U. Schöning, *Sparse sets, lowness and highness*, SIAM Journal on Computing, 15 (1986), pp. 739–746.

[5] R. Book, *On collapsing the polynomial-time hierarchy*, Information Processing Letters, 52 (1994), pp. 235–237.

[6] R. Book, T. Long, and A. Selman, *Quantitative relativizations of complexity classes*, SIAM Journal on Computing, 13 (1984), pp. 461–487.

[7] H. Buhrman, J. Kadin, and T. Thierauf, *On functions computable with nonadaptive queries to NP*, in Proceedings of the 9th Structure in Complexity Theory Conference, IEEE Computer Society Press, 1994, pp. 43–52.

[8] H. Buhrman, P. van Helden, and L. Torenvliet, *P-selective self-reducible sets: A new characterization of P*, in Proceedings of the 8th Structure in Complexity Theory Conference, IEEE Computer Society Press, May 1993, pp. 44–51.

[9] D. Denny-Brown, Y. Han, L. Hemaspaandra, and L. Torenvliet, *Semi-membership algorithms: Some recent advances*, SIGACT News, 25 (1994), pp. 12–23.

[10] S. Even and Y. Yacobi, *Cryptocomplexity and NP-completeness*, in Proceedings of the 7th International Colloquium on Automata, Languages, and Programming, Springer-Verlag *Lecture Notes in Computer Science*, 1980, pp. 195–207.

[11] S. Fenner, L. Fortnow, S. Kurtz, and L. Li, *An oracle builder's toolkit*, in Proceedings of the 8th Structure in Complexity Theory Conference, IEEE Computer Society Press, May 1993, pp. 120–131.

[12] S. Fenner, S. Homer, M. Ogiwara, and A. Selman, *On using oracles that compute values*, in Proceedings of the 10th Annual Symposium on Theoretical Aspects of Computer Science, Springer-Verlag *Lecture Notes in Computer Science #665*, Feb. 1993, pp. 398–407.

[13] R. Gavaldà, *Bounding the complexity of advice functions*, in Proceedings of the 7th Structure in Complexity Theory Conference, IEEE Computer Society Press, June 1992, pp. 249–254.

[14] R. Gavaldà and J. Balcázar, *Strong and robustly strong polynomial time reducibilities to sparse sets*, Theoretical Computer Science, 88 (1991), pp. 1–14.

[15] J. Grollmann and A. Selman, *Complexity measures for public-key cryptosystems*, SIAM Journal on Computing, 17 (1988), pp. 309–335.

[16] Y. Han, L. Hemaspaandra, and T. Thierauf, *Threshold computation and cryptographic security*, in Proceedings of the 4th International Symposium on Algorithms and Computation, Springer-Verlag *Lecture Notes in Computer Science #762*, Dec. 1993, pp. 230–239.

[17] E. Hemaspaandra, A. Naik, M. Ogiwara, and A. Selman, *P-selective sets, and reducing search to decision vs. self-reducibility*, Tech. Report 93-21, State University of New York at Buffalo, Department of Computer Science, Buffalo, NY, 1993.

[18] L. Hemaspaandra, A. Hoene, A. Naik, M. Ogiwara, A. Selman, T. Thierauf, and J. Wang, *Selectivity: Reductions, nondeterminism, and function classes*, Tech. Report TR-469, University of Rochester, Department of Computer Science, Rochester, NY, Aug. 1993.

[19] L. Hemaspaandra, A. Hoene, M. Ogiwara, A. Selman, T. Thierauf, and J. Wang, *Selectivity*, in Proceedings of the 5th International Conference on Computing and Information, IEEE Computer Society Press, 1993, pp. 55–59.

[20] C. Jockusch, *Semirecursive sets and positive reducibility*, Transactions of the AMS, 131 (1968), pp. 420–436.

[21] R. Karp and R. Lipton, *Some connections between nonuniform and uniform complexity classes*, in Proceedings of the 12th ACM Symposium on Theory of Computing, Apr. 1980, pp. 302–309. An extended version has also appeared as: Turing machines that take advice, *L'Enseignement Mathématique*, 2nd series 28, 1982, pages 191–209.

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

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

[24] J. Köbler and O. Watanabe, *New collapse consequences of NP having small circuits*, Tech. Report 94-11, Institut für Informatik, Universität Ulm, Ulm, Germany, Nov. 1994.

[25] M. Krentel, *The complexity of optimization problems*, Journal of Computer and System Sciences, 36 (1988), pp. 490–509.

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

[27] T. Long and M. Sheu, *A refinement of the low and high hierarchies*, Tech. Report OSU-CISRC-2/91-TR6, Ohio State University, Department of Computer Science, Columbus, Ohio, Feb. 1991.

[28] L. Longpré and A. Selman, *Hard promise problems and nonuniform complexity*, Theoretical Computer Science, 115 (1993), pp. 277–290.

[29] A. Meyer and M. Paterson, *With what frequency are apparently intractable problems difficult?*, Tech. Report MIT/LCS/TM-126, MIT Laboratory for Computer Science, Cambridge, MA, 1979.

[30] A. Meyer and L. Stockmeyer, *The equivalence problem for regular expressions with squaring requires exponential space*, in Proceedings of the 13th IEEE Symposium on Switching and Automata Theory, 1972, pp. 125–129.

[31] A. Naik, M. Ogiwara, and A. Selman, *P-selective sets, and reducing search to decision vs. self-reducibility*, in Proceedings of the 8th Structure in Complexity Theory Conference, IEEE Computer Society Press, May 1993, pp. 52–64.

[32] N. Nisan and A. Wigderson, *Hardness vs. randomness*, Journal of Computer and System Sciences, 49 (1994), pp. 149–167.

[33] J. Royer, Aug. 1993. Personal Communication.

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

[35] A. Selman, *Polynomial time enumeration reducibility*, SIAM Journal on Computing, 7 (1978), pp. 440–457.

[36] ———, *P-selective sets, tally languages, and the behavior of polynomial time reducibilities on NP*, Mathematical Systems Theory, 13 (1979), pp. 55–65.

[37] ———, *Promise problems complete for complexity classes*, Information and Computation, 78 (1988), pp. 87–98.

[38] ———, *A taxonomy of complexity classes of functions*, Journal of Computer and System Sciences, 48 (1994), pp. 357–381.

[39] L. Stockmeyer, *The polynomial-time hierarchy*, Theoretical Computer Science, 3 (1977), pp. 1–22.

[40] S. Toda, *PP is as hard as the polynomial-time hierarchy*, SIAM Journal on Computing, 20 (1991), pp. 865–877.

[41] 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.

[42] L. Valiant and V. Vazirani, *NP is as easy as detecting unique solutions*, Theoretical Computer Science, 47 (1986), pp. 85–93.

[43] O. Watanabe and S. Toda, *Structural analysis of the complexity of inverse functions*, Mathematical Systems Theory, 26 (1993), pp. 203–214.

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