Complements of Multivalued Functions

Stephen Fenner *
University of Southern Maine

Frederic Green †
Clark University

Steven Homer ‡
Boston University

Alan L. Selman §
State University of New York at Buffalo

Thomas Thierauf ¶
Universität Ulm

Heribert Vollmer ‖
Universität Würzburg

Abstract

We study the class coNPMV of complements of NPMV functions. Though defined symmetrically to NPMV this class exhibits very different properties. We clarify the complexity of coNPMV by showing that, surprisingly, it is essentially the same as that of $\mathrm{NPMV}^{\mathrm{NP}}$. Complete functions for coNPMV are exhibited and central complexity-theoretic properties of this class are studied. We show that computing maximum satisfying assignments can be done in coNPMV, which leads us to a comparison of NPMV and coNPMV with Krentel's classes Max P and Min P. The difference hierarchy for NPMV is related to the query hierarchy for coNPMV. Finally, we examine a functional analogue of Chang and Kadin's relationship between a collapse of the Boolean hierarchy over NP and a collapse of the polynomial time hierarchy.

*Computer Science Department, University of Southern Maine, Portland, ME 04103. E-mail: fenner@cs.usm.maine.edu. Supported in part by the NSF under grants CCR 92-09833 and CCR 95-01794.

†Department of Mathematics and Computer Science, Clark University, Worcester, MA 01610. E-mail: fgreen@black.clarku.edu.

‡Computer Science Department, Boston University, Boston, MA 02215. E-mail: homer@cs.bu.edu. Supported in part by the NSF under grant NSF-CCR-9400229.

§Department of Computer Science, 226 Bell Hall, State University of New York at Buffalo, Buffalo, NY 14260. E-mail: selman@cs.buffalo.edu. Supported in part by the NSF under grant NSF-CCR-9400229.

¶Abteilung Theoretische Informatik, Universität Ulm, Oberer Eselsberg, 89069 Ulm, Germany. Email: thierauf@informatik.uni-ulm.de.

‖Theoretische Informatik, Universität Würzburg, Am Exerzierplatz 3, D-97072 Würzburg, Germany. E-mail: vollmer@informatik.uni-wuerzburg.de. Work supported by the Alexander von Humboldt foundation under a Feodor Lynen scholarship.

1. Introduction

Consider the complexity class NPMV of partial multivalued functions that are computed nondeterministically in polynomial time. As this class captures the complexity of computing witnesses of sets in NP, by studying this class, and more generally, by studying relations between such complexity classes of partial multivalued functions, we directly contribute to understanding the complexity of computing witnesses. It is well-known that a partial multivalued function f belongs to NPMV if and only if it is polynomial length-bounded and $\text{graph}(f) = \{ \langle x, y \rangle \mid y \text{ is a value of } f(x) \}$ belongs to NP.

Now consider the class coNPMV. We will give a formal definition in the preliminary section below. For now, let us state that a partial multivalued function f belongs to coNPMV if and only if it is polynomial length-bounded and $\text{graph}(f)$ belongs to coNP. Given this symmetry, graphs of functions in NPMV are in NP while graphs of functions in coNPMV are in coNP, and given what we know about NP and coNP, one might expect that coNPMV has essentially the same complexity as NPMV. Indeed, it is easy to see that coNPMV = NPMV if and only if NP = coNP. However, the point of this paper is to show that in many ways coNPMV is a more powerful class than is NPMV. One can derive more information from computing the complement of a function in NPMV than from computing the function. For one example of this phenomena, we prove here that coNPMV is not included in FP^{NPMV} unless the polynomial hierarchy collapses. (This is an extension of a result of Fenner *et al.* [FHOS93].) Yet, it is obvious that coNPMV can be computed in polynomial time with one query to coNPMV. Thus, a coNPMV oracle provides more information than an NPMV oracle. This is surprising, for function oracles, just as set oracles, provide knowledge about both their domains and their co-domains.

We will define many-one reductions between multivalued functions, (This will be a straightforward adaptation of the many-one metric reducibility of Krentel [Kre88].) and we will consider many-one complete functions for NPMV and coNPMV. We will see that functions such as *sat*, defined so that y is a value of $sat(x)$ if and only if y is a satisfying assignment of x, which clearly is a complete function for NPMV, is also in the class coNPMV. Even the seemingly more powerful function *maxsat* is in coNPMV. (Recall that *maxsat* is complete for FP^{NP}.) Clearly, these function classes have strange closure properties, which we describe below.

Our results suggest that the complexity of coNPMV is essentially the same as the complexity of NPMV^{NP}. We prove that $\text{NPMV}^{\text{NP}} = \{\pi_2^1\} \circ \text{coNPMV}$ (where $\{\pi_2^1\}$ is the projection function that maps a pair of strings to its first component) and we prove that NPMV^{NP} is the many-one closure of coNPMV. In addition, we prove that

$$\text{coNPMV} \subseteq \text{NPMV}(2) \subseteq \cdots \subseteq \text{NPMV}(k) \subseteq \cdots \subseteq \text{NPMV}(n^{O(1)}) \subseteq \text{NPMV}^{\text{NP}},$$

where $\text{NPMV}(k)$ is the k-th level of the difference hierarchy for NPMV as defined by Fenner *et al.* [FHOS93]. The difference hierarchy over NPMV does not collapse unless the polynomial hierarchy collapses. This is striking. Even though the complexity of coNPMV is essentially the same as that of NPMV^{NP}, there is an infinite hierarchy of complexity classes between them. This is also supported by the following: We show that if the difference hierarchy for NPMV collapses, then the NPMV oracle hierarchy collapses. This result is the functional analogue of the well-known result by Chang and Kadin relating a collapse of the Boolean hierarchy over NP to a collapse of the polynomial time hierarchy.

2. Preliminaries

We fix Σ to be the finite alphabet $\{0,1\}$. Let $<$ denote the standard lexicographic order on Σ^*.

Let f be a relation on $\Sigma^* \times \Sigma^*$. We will call f a *(partial) multivalued function* from Σ^* to Σ^*. By $f(x) \mapsto y$ we denote that $(x,y) \in f$ and say that f *maps x to y*. By set-$f(x)$ we denote the set of outcomes of f on x, set-$f(x) = \{\, y \mid f(x) \mapsto y \,\}$. The *graph of* f is $\mathrm{graph}(f) = \{\, \langle x,y \rangle \mid f(x) \mapsto y \,\}$. The *domain of* f, $\mathrm{dom}(f)$, is the set of x where set-$f(x)$ is nonempty. We will say that f is undefined at x if $x \notin \mathrm{dom}(f)$. The domain of a class $\mathcal{F}$ of functions is $\mathrm{dom}(\mathcal{F}) = \bigcup_{f\in\mathcal{F}} \mathrm{dom}(f)$.

Given partial multivalued functions f and g, define g to be a *refinement of* f if $\mathrm{dom}(g) = \mathrm{dom}(f)$ and for all x, we have set-$g(x) \subseteq$ set-$f(x)$. Let $\mathcal{F}$ and $\mathcal{G}$ be classes of partial multivalued functions. Purely as a convention, if f is a partial multivalued function, we define $f \in_c \mathcal{G}$ if $\mathcal{G}$ contains a refinement of f, and we define $\mathcal{F} \subseteq_c \mathcal{G}$ if for every $f \in \mathcal{F}$, $f \in_c \mathcal{G}$. This notation is consistent with our intuition that $\mathcal{F} \subseteq_c \mathcal{G}$ should entail that the complexity $\mathcal{F}$ is not greater than the complexity of $\mathcal{G}$.

A transducer T is a nondeterministic Turing machine with a read-only input tape, a write-only output tape, and accepting states in the usual manner. T computes a value y on an input string x if there is an accepting computation of T on x for which y is the final content of T's output tape. (In this case, we will write $T(x) \mapsto y$.) Such transducers compute partial, multivalued functions. (As transducers do not typically accept all input strings, when we write "function", "partial function" is always intended. If a function f is total, it will always be explicitly noted.)

The following classes of partial functions were first defined by Book, Long, and Selman [BLS84].

- NPMV is the set of all partial, multivalued functions computed by nondeterministic polynomial time-bounded transducers;
- NPSV is the set of all $f \in$ NPMV that are single-valued;
- FP is the set of all partial functions computed by deterministic polynomial time-bounded transducers.

A function f belongs to NPMV if and only if it is polynomially length-bounded and *graph*(f) belongs to NP. The domain of every function in NPMV belongs to NP.

Fenner *et al.* [FHOS93] define the *difference hierarchy over* NPMV as follows.

Let $\mathcal{F}$ be a class of partial multivalued functions. A partial multivalued function f is in co$\mathcal{F}$ if there exist $g \in \mathcal{F}$ and a polynomial p such that for every x,

- set-$f(x) \;=\; \Sigma^{p(|x|)} -$ set-$g(x)$.

Let $\mathcal{F}$ and $\mathcal{G}$ be two classes of partial multivalued functions. A partial multivalued function h is in $\mathcal{F} \wedge \mathcal{G}$, respectively $\mathcal{F} \vee \mathcal{G}$, if there exist partial multivalued functions $f \in \mathcal{F}$ and $g \in \mathcal{G}$ such that for every x,

- set-$h(x) \;=\;$ set-$f(x) \cap$ set-$g(x)$, respectively set-$h(x) \;=\;$ set-$f(x) \cup$ set-$g(x)$.

Let $\mathcal{F} - \mathcal{G}$ denote $\mathcal{F} \wedge \mathrm{co}\mathcal{G}$. Then, NPMV$(k)$ is the class of partial multivalued functions defined in the following way:

1. NPMV$(1) =$ NPMV, and
2. for $k \geq 2$, NPMV$(k) =$ NPMV $-$ NPMV$(k-1)$.

Fenner *et al.* prove that for every $k \geq 1$, $f \in \mathrm{NPMV}(k)$ if and only if f is polynomially length-bounded and $graph(f) \in \mathrm{NP}(k)$.

In particular we are interested in the class coNPMV. It follows that a function f belongs to coNPMV if and only if it is polynomially length-bounded and $graph(f)$ belongs to coNP. Observe that the classes NPMV and coNPMV satisfy the nice symmetry that graphs of functions in the former class are in NP and of the latter class are in coNP.

The class $\mathrm{FP}^{\mathrm{NP}}$ is the collection of partial functions computed in polynomial time with oracles in NP.

The primary new contribution of Fenner *et al.* is the development of hierarchies of classes of functions that access classes of partial functions as oracles. This development is based on the following description of oracle Turing machines with oracles that compute partial functions. Assume first that the oracle is a single-valued partial function. Let $\perp$ be a symbol not belonging to the finite alphabet Σ. In order for a machine M to access a partial function oracle, M contains a write-only input oracle tape, a separate read-only output oracle tape, and a special oracle call state q. When M enters state q, if the string currently on the oracle input tape belongs to the domain of the oracle partial function, then the result of applying the oracle appears on the oracle output tape, and if the string currently on the oracle input tape does not belong to the domain of the oracle partial function, then the symbol $\perp$ appears on the oracle output tape. Thus, if the oracle is some partial function g, given an input x to the oracle, the oracle, if called, returns a value $g(x)$ if one exists, and returns $\perp$ otherwise. (It is possible that M may read only a portion of the oracle's output if the oracle's output is too long to read with the resources of M.) We shall assume, without loss of generality, that M never makes the same oracle query more than once, i.e., all of M's queries (on any possible computation path) are distinct.

If g is a single-valued partial function and M is a deterministic oracle transducer as just described, then we let $M[g]$ denote the single-valued partial function computed by M with oracle g.

2.1 Definition. *[FHOS93] Let f and g be multivalued partial functions. f* is Turing reducible to *g* in polynomial time, *$f \leq_{\mathrm{T}}^{\mathrm{P}} g$, if for some deterministic oracle transducer M, for every single-valued refinement g' of g, $M[g']$ is a single-valued refinement of f.*

Fenner *et al.* prove that $\leq_{\mathrm{T}}^{\mathrm{P}}$ is a reflexive and transitive relation over the class of all partial multivalued functions.

Let $\mathcal{F}$ be a class of partial multivalued functions. $\mathrm{FP}^{\mathcal{F}}$ denotes the class of partial multivalued functions f that are $\leq_{\mathrm{T}}^{\mathrm{P}}$-reducible to some $g \in \mathcal{F}$. $\mathrm{FP}^{\mathcal{F}[k]}$ (respectively, $\mathrm{FP}^{\mathcal{F}[\log]}$) denotes the class of partial multivalued functions f that are $\leq_{\mathrm{T}}^{\mathrm{P}}$-reducible to some $g \in \mathcal{F}$ via a machine that, on input x, makes k adaptive queries (respectively, $\mathcal{O}(\log |x|)$ adaptive queries) to its oracle.

In particular, these definition templates define such classes of multivalued partial functions as $\mathrm{FP}^{\mathrm{NPMV}}$, and $\mathrm{NPMV}^{\mathrm{NPMV}}$.

We will use the following generalization of the many-one metric reducibility of Krentel [Kre88] in order to discuss complete functions for classes of multivalued functions.

2.2 Definition. Given partial multivalued functions $f, g : \Sigma^* \to \Sigma^*$, we say f *is metric many-one reducible to* g, or symbolically, $f \leq_{\mathrm{m}}^{\mathrm{P}} g$, if there are functions $t_1, t_2 \in \mathrm{FP}$ such that for any x,

$$t_2(x, g \circ t_1(x)) \mapsto y \Longrightarrow f(x) \mapsto y. \tag{1}$$

(That is to say, $t_2(id, g \circ t_1)$ is a refinement of f.)

The motivation underlying this definition is that, given a polynomial-time algorithm for computing a value of g (i.e., a single-valued refinement thereof), the reduction should yield a polynomial-time algorithm for computing a value of f. Definition 2.2 gives the most general formulation of this idea. An alternative stronger definition would be to insist that $t_2(x, g \circ t_1(x))$ and $f(x)$ are *equal*. Obviously, $f \leq_{\mathrm{m}}^{\mathrm{P}} g$ implies $f \leq_{\mathrm{T}}^{\mathrm{P}} g$.

The classes that we have been considering relate in interesting ways to studies of the complexity of optimization problems. In order to capture the complexity of optimization problems Krentel [Kre88] defined the complexity classes MaxP and MinP to contain the functions computable by taking the maximum, minimum, respectively, over sets of feasible solutions of problems in NP. Further, Krentel extended these classes to hierarchies of classes of optimization functions [Kre92]. These functions are best understood using Krentel's notion of *metric Turing machine*, which we now recall. Consider nondeterministic polynomial time Turing machines that print an output value on every path such that with every inner node of the computation tree either the function min or the function max is associated (for the classes MinP and MaxP, resp.). Evaluating such a machine's computation tree in the obvious bottom-up manner thus yields, for each input string, a uniquely defined integer value. Thus, metric Turing machines define functions from input words to integers.

Then, Krentel defined the class Σ_k^{MM} to consist of all functions that can be computed by machines described above that on every path make at most $k-1$ alternations between min and max states where the initial state is a max state. Vollmer and Wagner [VW93, VW95] gave a detailed structural examination of Krentel's hierarchy, as did Ramachandran [Ram94]. Here, we just define classes MaxP and MinP, using an operator-characterization from [VW95].

$$\begin{array}{lll} h \in \mathrm{MaxP} & \Longleftrightarrow \exists f,\ g \in \mathrm{FP} & h(x) = \max_{0 \leq y \leq g(x)} f(x,y), \\ h \in \mathrm{MinP} & \Longleftrightarrow \exists f,\ g \in \mathrm{FP} & h(x) = \min_{0 \leq y \leq g(x)} f(x,y). \end{array}$$

The class NPMV consists of partial multivalued functions, while MaxP consists of total single-valued functions. We compare the complexities of classes of multivalued functions (with one another, or with classes of single-valued functions) by using the relation "$\subseteq_c$". But we still have to face the problem that MaxP and MinP are classes of total functions and NPMV is not. To make a reasonable comparison possible, we thus introduce the following natural partial counterparts of Krentel's classes.

Consider a polynomial-time metric Turing machine, but now allow the machine to output a special symbol $\perp$ that denotes that the computation on the corresponding path ends with an undefined result. Then evaluate the computation tree in the obvious way, using the equations $\max(x, \perp) = \max(\perp, x) = \min(x, \perp) = \min(\perp, x) = x$, for all x (including $\perp$ itself). The function defined by the machine is undefined on input x, if the just sketched evaluation results in the value $\perp$ (which is the case if all paths result in $\perp$). (Equivalently, we could of course give an operator characterization as above.) In the following, when talking about MaxP or MinP, we will always mean the just described partial versions. Since these analogues look very natural we decided not to introduce new notation for these classes but to stick to the original names MaxP and MinP.

3. Reducibilities and Complete Functions

NPMV is precisely the class of functions that compute witnesses for NP sets in the following sense. For any NP set L there exist a set $A \in \mathrm{P}$ and a polynomial p such that for all x, we have $x \in L \Longleftrightarrow \exists y \in \Sigma^{p(|x|)} :\ (x,y) \in A$. Any y such that $(x,y) \in A$ is called a *witness for* x (with respect to A). Clearly, there is a NPMV function f such that set-$f(x)$ is exactly the set of witnesses for x. On the other hand, any NPMV function f defines a NP set as the set of all x such that there exists an output of f on x. In other words, $\mathrm{dom}(f) \in \mathrm{NP}$. As a consequence of this discussion, we see that $\mathrm{dom}(\mathrm{NPMV}) = \mathrm{NP}$.

Next, we extend the notion of a witness to Σ_2^p. For any Σ_2^p set L there exist a set $B \in \mathrm{coNP}$ and a polynomial p such that for all x, we have $x \in L \iff \exists y \in \Sigma^{p(|x|)} : (x, y) \in B$. A y such that $(x, y) \in B$ is called a *witness for* x (with respect to B). What function class captures the computation of witnesses for Σ_2^p sets? Since $\Sigma_2^p = \mathrm{NP}^{\mathrm{NP}}$, certainly witnesses can be computed in $\mathrm{NPMV}^{\mathrm{NP}}$. However, we will see below that a seemingly weaker class already suffices to do so.

Let us consider set L again. Since $B \in \mathrm{coNP}$, there is a set $A \in \mathrm{P}$ such that $(x, y) \in B \iff \forall z \in \Sigma^{p(|x|)} : (x, y, z) \in A$. Now consider the following multivalued function f.

$$f(x) \mapsto y \in \Sigma^{p(|x|)} \iff \exists z \in \Sigma^{p(|x|)} : (x, y, z) \notin A.$$

Clearly, $f \in \mathrm{NPMV}$ and set-$f(x)$ is precisely the set of *nonwitnesses* for x; that is, the set of witnesses for x equals $\Sigma^{p(|x|)}$ − set-$f(x)$. Hence, coNPMV can compute witnesses for sets in Σ_2^p. Conversely, for any coNPMV function f, we have $\mathrm{dom}(f) \in \Sigma_2^p$. This is because, for any x, $x \in \mathrm{dom}(f) \iff \exists y \in \Sigma^{p(|x|)} : y \in$ set-$f(x)$. Thus, coNPMV is *precisely* the class of functions that computes witnesses for Σ_2^p sets. As a consequence, we have the following proposition.

3.1 Proposition. $\mathrm{dom}(\mathrm{coNPMV}) = \Sigma_2^p$.

Witnesses of Σ_2^p complete sets can give rise to complete functions for coNPMV. Consider for example the satisfiability problem QBF_2 for Boolean formulas with two quantifiers

$$\varphi(x_1, \ldots, x_k, y_1, \ldots, y_l) \in \mathrm{QBF}_2 \iff \exists x_1, \ldots, x_k \, \forall y_1, \ldots, y_l : \ \varphi(x_1, \ldots, x_k, y_1, \ldots, y_l) = 1.$$

Let F_2 be the multivalued function that computes witnesses, i.e., partial assignments $(x_1, \ldots, x_k)$, for QBF_2 formulas φ as above.

3.2 Theorem. *F_2 is $\leq_{\mathrm{m}}^{\mathrm{P}}$-complete for* coNPMV.

Proof. We have argued already that $F_2 \in \mathrm{coNPMV}$. Let f be any coNPMV function. There is a NP transducer M and a polynomial p such that for all x, we have set-$f(x) = \Sigma^{p(|x|)}$ − set-$M(x)$. We show how to compute a $y \in$ set-$f(x)$ from $F_2(\varphi_x)$, for an appropriately constructed formula φ_x.

Define a machine M' on input x as follows. First, M' guesses a $y \in \Sigma^{p(|x|)}$. Then, M' simulates M on input x. If M outputs y on the simulated path, then M' rejects. Otherwise, M' accepts.

We apply Cook's reduction to x with M' as the underlying machine. This will give a Boolean formula φ_x that, intuitively, describes the work of M' on input x. The variables of φ_x can be partitioned into two parts:

- say $y_1, \ldots, y_k$, that are used to describe that M' guesses a $y \in \Sigma^{p(|x|)}$, and
- say $z_1, \ldots, z_l$, that are used to describe the subsequent simulation of M.

Furthermore, from any setting of the variables $y_1, \ldots, y_k$ of φ_x, we can reconstruct in polynomial time the $y \in \Sigma^{p(|x|)}$ guessed by M'.

Let us fix a setting of the variables $y_1, \ldots, y_k$ and let $y \in \Sigma^{p(|x|)}$ be the corresponding string guessed by M'. Then we have

$$\begin{aligned} \forall z_1, \ldots, z_l : \ \varphi_x(y_1, \ldots, y_k, z_1, \ldots, z_l) = 1 \quad &\iff \quad M' \text{ accepts on all paths following } y \\ &\iff \quad y \notin \text{set-}M(x) \\ &\iff \quad f(x) \mapsto y \end{aligned}$$

□

A crucial point in the above proof is that Cook's reduction maintains witnesses. That is, from a given assignment for the constructed formula φ_x one can recover a corresponding path of the nondeterministic machine. Thus any Σ_2^p complete set sharing this property with QBF_2 defines a coNPMV complete function in an analogous way. Note moreover that there is a one-one correspondence between witnesses and assignments (that is computable and invertible in polynomial time).

As an example, consider the following set. For any NPMV function f and some polynomial p,

$$L_f \;=\; \{\, x \mid (\exists y,\ |y| \le p(|x|))(\ \forall z,\ |z| \le p(|x|))\ f(x) \not\mapsto zy \,\}.$$

Clearly, for every $f \in$ NPMV, we have that L_f is in Σ_2. Thus, in particular L_{sat} is in Σ_2, is even Σ_2 complete and has the above mentioned property. We conclude that the corresponding witness function, $\overline{pre}(sat)$, is complete for coNPMV, where

$$\overline{pre}(sat)(\varphi) \mapsto y \iff |y| \le p(|x|)\ \&\ (\ \forall z,\ |z| \le p(|\varphi|))\ sat(\varphi) \not\mapsto zy.$$

3.3 Theorem. *$\overline{pre}(sat)$ is $\le_{\mathrm{m}}^{\mathrm{P}}$-complete for* coNPMV.

NPMV is not closed under $\le_{\mathrm{m}}^{\mathrm{P}}$-reductions; in fact, it is possible to have $g \in$ NPMV and $f \le_{\mathrm{m}}^{\mathrm{P}} g$ but f be nonrecursive. (For example, take $f(x,y)$ to map to 1 if Turing machine x halts on input x, 0 if it does not, and (in either case) $z11$ if the string z is a satisfying assignment of Boolean formula y. Since $sat(x,y)$ (defined to equal $sat(y)$, ignoring the input x) is a refinement of f, $f \le_{\mathrm{m}}^{\mathrm{P}} sat$.) However, NPMV is closed under this reduction in a weaker sense, defined below.

3.4 Definition. We say a class $\mathcal{C}$ is *c-closed* under reducibility $\le_r$ if, $g \in \mathcal{C}$ and $f \le_r g$ implies $f \in_c \mathcal{C}$.

It is immediate from this definition that NPMV is c-closed under $\le_{\mathrm{m}}^{\mathrm{P}}$. One might suspect that this same fact holds for coNPMV by "complementing" f and g. (Here the complement of a function h is given by the definition on page 2 as a function whose values, on input x, are $\Sigma^{p(|x|)} -$ set-$h(x)$.) However, the closure does not follow as in the definition of metric reduction (Definition 2.2), since there we do not insist that *nonoutputs* of $g(t_1(x))$ be transformed by t_2 into *nonoutputs* of $f(x)$. In fact, it is quite unlikely that coNPMV is c-closed under this reducibility. If it were, since $sat \in$ coNPMV and sat is complete for NPMV, we would get that NPMV $\subseteq_c$ coNPMV. But this seems to be very unlikely as the following extension of a result of Fenner *et al.* [FHOS93] shows.

3.5 Theorem. NPMV $\subseteq$ coNPMV $\iff$ NPMV $\subseteq_c$ coNPMV $\iff$ NP = coNP.

Proof. We cycle through the implications. The first implication is trivial. For the second, let $L \in$ NP. Define function

$$\chi_L(x) \;=\; \begin{cases} 1 & \text{if } x \in L \\ \perp & \text{otherwise .} \end{cases}$$

Then we have $\chi_L \in$ NPMV, and hence, by assumption, $\chi_L \in$ coNPMV. Therefore, graph$(\chi_L) \in$ coNP, which implies that $L \in$ coNP since $x \in L \iff (x,1) \in$ graph(χ_L).

Now suppose that NP = coNP and let $f \in$ NPMV. Then graph$(f) \in$ NP, and therefore in coNP by assumption. Define function g as $g(x) \mapsto y \iff \langle x,y\rangle \notin$ graph(f), where $y \in \Sigma^{p|x|)}$, for some polynomial p. Then we have $g \in$ NPMV and set-$f(x) = \Sigma^{p|x|)} -$ set-$g(x)$. $\square$

We observe that the proof of Theorem 3.5 shows also that NPSV $\subseteq$ coNPMV $\iff$ NP = coNP, even though it is fairly easy to see that NPSV$_t$, the class of all total NPSV functions, *is* contained in coNPMV. We also note that Theorem 3.5 extends to higher levels of the difference hierarchies over NPMV and NP, that is NPMV(k) $\subseteq$ coNPMV(k) $\iff$ NPMV(k) $\subseteq_c$ coNPMV(k) $\iff$ NP(k) = coNP(k).

3.6 Corollary. coNPMV *is c-closed under* $\leq_{\mathrm{m}}^{\mathrm{P}}$*-reducibility if and only if* NP = coNP.

Hence we can have the situation that a function that is complete for some class, like *sat* for NPMV, is in coNPMV without the corresponding class being contained in coNPMV. This seems to happen again for *maxsat*. Fenner *et al.* [FHOS93] show that *maxsat* $\in$ NPMV(2). In fact, it is even in coNPMV. However, we will show in the next section that the corresponding classes, namely MaxP or FP$^{\mathrm{NP}}$ are unlikely to be contained in coNPMV.

3.7 Theorem. *maxsat* $\in$ coNPMV.

Proof. Consider an NPMV machine M that, on input of a formula φ, guesses an assignment y for φ. If y does not satisfy φ, then M outputs y. Otherwise, if y does satisfy φ, M guesses another assignment $y' > y$. If y' also satisfies φ, M outputs y, otherwise M rejects.

M outputs every assignment except the maximum satisfying one (if there is one). Hence *maxsat* $\in$ coNPMV. □

Recall the theorem of Krentel, which states FP$^{\mathrm{NP}}$ = FP$^{\mathrm{MaxP[1]}}$. Since *maxsat* is complete for MaxP (regarding MaxP as a class of partial functions), this implies FP$^{\mathrm{NPMV}}$ = FP$^{\mathrm{NP}}$ = FP$^{\mathrm{MaxP[1]}}$, so that FP$^{\mathrm{NPMV}}$ = FP$^{\mathrm{coNPMV[1]}}$. As we will see in the next section, coNPMV is a more powerful class than NPMV, as this relation suggests.

4. Properties of coNPMV

In this section, we will relate coNPMV to other function classes. As we have already seen at the end of the preceding section, coNPMV seems to be a more powerful class than NPMV. This is somewhat surprising in light of the aforementioned symmetry in the definitions of coNPMV and NPMV by their graphs.

The following theorem shows that coNPMV is in fact very close to NPMV$^{\mathrm{NP}}$. Let π_2^1 denote the projection function that maps a pair of strings to its first component.

4.1 Theorem. NPMV$^{\mathrm{NP}}$ = $\{\pi_2^1\} \circ$ coNPMV.

Proof. To prove the right to left containment, note first that coNPMV $\subseteq$ NPMV$^{\mathrm{NP}}$, since an NPMV$^{\mathrm{NP}}$ machine can guess the value of a coNPMV function and verify it is correct with a query to coNP. Now observe that the projection of any NPMV$^{\mathrm{NP}}$ function is still in NPMV$^{\mathrm{NP}}$, whence $\{\pi_2^1\} \circ$ coNPMV $\subseteq$ NPMV$^{\mathrm{NP}}$.

For the other direction, let $f \in$ NPMV$^{\mathrm{NP}}$. Then by a standard argument there is a polynomial q and a predicate $R \in$ P such that for any x and y such that $|y| \leq q(|x|)$,

$$f(x) \mapsto y \text{ iff } (\exists z, |z| \leq q(|x|))(\forall w, |w| \leq q(|x|))R(x, y, z, w).$$

Define f' such that for any x and y, z where $|y|, |z| \leq q(|x|)$,

$$f'(x) \mapsto \langle y, z\rangle \text{ iff } (\forall w, |w| \leq q(|x|))R(x, y, z, w).$$

So $\neg R$ witnesses that $f' \in$ coNPMV. But $f(x) = \pi_2^1 \circ f'(x)$, which shows that $f \in \{\pi_2^1\} \circ$ coNPMV. □

There is a counterintuitive facet to Theorem 4.1. The reason why it is likely that coNPMV is a proper subclass of $\mathrm{NPMV}^{\mathrm{NP}}$ is not because outputs of coNPMV functions give too little information, but rather that they give too much. We can compute an arbitrary $\mathrm{NPMV}^{\mathrm{NP}}$ function simply by throwing away part of the output of a coNPMV function. This is what the projection operator accomplishes, and it is most likely necessary.

Applying Theorem 4.1, many properties of $\mathrm{NPMV}^{\mathrm{NP}}$ now carry over to coNPMV. In the previous section we have shown function F_2 and $\overline{pre}(sat)$ complete for coNPMV. Since the projection function is in FP, we get that those functions are complete for $\mathrm{NPMV}^{\mathrm{NP}}$ as well.

4.2 Corollary. $\mathrm{NPMV}^{\mathrm{NP}}$ *is the c-closure of* coNPMV *under* $\leq_{\mathrm{m}}^{\mathrm{P}}$*-reducibility.*

In particular, we get

4.3 Corollary. $\mathrm{FP}^{\mathrm{coNPMV}[1]} = \mathrm{FP}^{\mathrm{NPMV}^{\mathrm{NP}}[1]}$ *and* $\mathrm{FP}^{\mathrm{coNPMV}} = \mathrm{FP}^{\mathrm{NPMV}^{\mathrm{NP}}} = \mathrm{F}\Delta_3^{\mathrm{P}}$.

Observe by contrast that $\mathrm{FP}^{\mathrm{NPMV}} = \mathrm{FP}^{\mathrm{NP}} = \mathrm{FP}^{\mathrm{Min\,P}} = \mathrm{FP}^{\mathrm{Max\,P}}$, so coNPMV and NPMV define different Δ-levels of the functional PH.

Fenner *et al.* [FHOS93] have shown that $\mathrm{NPMV}(2) \subseteq \mathrm{FP}^{\mathrm{NPMV}} \iff \Sigma_2^p = \Delta_2^p$ Note that in contrast for the corresponding language classes we have $\mathrm{NP}(k) \subseteq \mathrm{FP}^{\mathrm{NP}}$ for all k. We can now improve the result of Fenner *et al.*

4.4 Corollary. $\mathrm{coNPMV} \subseteq \mathrm{FP}^{\mathrm{NPMV}} \iff \Sigma_2^p = \Delta_2^p$.

Proof. If $\Sigma_2^p = \Delta_2^p$, then

$$\mathrm{coNPMV} \subseteq \mathrm{FP}^{\mathrm{coNPMV}} = \mathrm{FP}^{\mathrm{NPMV}^{\mathrm{NP}}} = \mathrm{FP}^{\Sigma_2^p} = \mathrm{FP}^{\Delta_2^p} = \mathrm{FP}^{\mathrm{NP}} = \mathrm{FP}^{\mathrm{NPMV}},$$

where the last equality is Theorem 1 in the paper by Fenner *et al.* [FHOS93] and the second follows from the relativized version of the same theorem. Conversely, if $\mathrm{coNPMV} \subseteq \mathrm{FP}^{\mathrm{NPMV}}$, then $\mathrm{dom}(\mathrm{coNPMV}) \subseteq \mathrm{P}^{\mathrm{NP}} = \Delta_2^p$, so that $\Sigma_2^p \subseteq \Delta_2^p$. □

Now we will compare the query hierarchy for coNPMV with the difference hierarchy for NPMV. We start with the following:

4.5 Theorem. $\mathrm{coNPMV} \subseteq \mathrm{NPMV}(2) \subseteq \cdots \subseteq \mathrm{NPMV}(k) \subseteq \cdots \subseteq \mathrm{NPMV}(n^{O(1)}) \subseteq \mathrm{NPMV}^{\mathrm{NP}}$. *All of the inclusions are strict unless the polynomial time hierarchy collapses.*

Proof. Fenner *et al.* [FHOS93] showed that $f \in \mathrm{NPMV}(k)$ if and only if f's graph is in $\mathrm{NP}(k)$, and $f \in \mathrm{coNPMV}$ if and only if f's graph is in coNP. It is easy to see that the same holds for $\mathrm{NPMV}(n^{O(1)})$ and $\mathrm{NP}(n^{O(1)})$. The only non-trivial inclusion is the last one. So let $f \in \mathrm{NPMV}(n^{O(1)})$. Then the graph of f is in $\mathrm{NP}(n^{O(1)})$, which is known to be equal to $\mathrm{P}^{\mathrm{NP}}[\log]$ [Wag90]. Obviously, f can be computed by an NPMV algorithm with access to a $\mathrm{P}^{\mathrm{NP}}[\log]$ oracle: simply guess an output of f and querying its graph check that the guess is correct. Thus, $\mathrm{NPMV}(n^{O(1)}) \subseteq \mathrm{NPMV}^{\mathrm{P}^{\mathrm{NP}}[\log]} \subseteq \mathrm{NPMV}^{\mathrm{NP}}$. This establishes the inclusions.

Suppose $\mathrm{coNPMV} = \mathrm{NPMV}(2)$. Take a set $A \in \mathrm{NP}(2)$. Interpret A as a graph of a function $f \in \mathrm{NPMV}(2)$. By assumption, $f \in \mathrm{coNPMV}$, thus $A \in \mathrm{coNP}$. Thus, we have $\mathrm{NP}(2) \subseteq \mathrm{coNP}$, which implies the collapse of the polynomial time hierarchy [Kad88]. The proofs for the other inclusions are completely analogous. □

Theorem 4.5 is striking. Even though coNPMV is "very close" to $\mathrm{NPMV}^{\mathrm{NP}}$, there is an infinite hierarchy of complexity classes between them.

4.6 Theorem. $\mathrm{NPMV}^{\mathrm{NP}} \subseteq \mathrm{FP}^{\mathrm{coNPMV}[1]} \subseteq \cdots \subseteq \mathrm{FP}^{\mathrm{coNPMV}[k]} \subseteq \cdots \subseteq \mathrm{FP}^{\mathrm{coNPMV}} = \mathrm{F}\Delta_3^{\mathrm{P}}$. *All inclusions are strict unless the polynomial time hierarchy collapses.*

Proof. The first inclusion and the equality follow from Corollary 4.3. All other inclusions are trivial.

Suppose $\mathrm{FP}^{\mathrm{coNPMV}[1]} \subseteq \mathrm{NPMV}^{\mathrm{NP}}$. This is equivalent to $\mathrm{FP}^{\mathrm{NPMV}^{\mathrm{NP}}[1]} \subseteq \mathrm{NPMV}^{\mathrm{NP}}$, which implies $\mathrm{P}^{\Sigma_2^p[1]} \subseteq \Sigma_2^p$. But then $\Pi_2^p = \Sigma_2^p$, and $\mathrm{PH} = \Sigma_2^p$. For the other inclusions, suppose $\mathrm{FP}^{\mathrm{coNPMV}[k]} = \mathrm{FP}^{\mathrm{coNPMV}[k+1]}$. Then $\mathrm{FP}^{\mathrm{NPMV}^{\mathrm{NP}}[k]} = \mathrm{FP}^{\mathrm{NPMV}^{\mathrm{NP}}[k+1]}$. By a theorem of Fenner *et al.* [FHOS93], this implies that $\mathrm{FP}^{\Sigma_2^p[k]} = \mathrm{FP}^{\Sigma_2^p[k+1]}$, which, by a relativization of Kadin's theorem [Kad88], implies that the polynomial hierarchy collapses. □

Thus we see, combining Theorems 4.5 and 4.6, that all classes of the difference hierarchy over NPMV are included in the query hierarchy over coNPMV, in fact already in its first level. There are (under reasonable assumptions) no inclusions in the opposite direction. Concerning the relationship between the query hierarchy over NPMV and the difference hierarchy over NPMV, we know from Fenner *et al.* [FHOS93] that all classes of the first hierarchy are included in certain classes of the second hierarchy. Any inclusion in the opposite direction implies $\mathrm{coNPMV} \subseteq \mathrm{FP}^{\mathrm{NPMV}}$, which again implies a collapse of the polynomial time hierarchy, by Corollary 4.4.

We now turn to the relationship between the complexity of computing maximal and minimal witnesses (as in MaxP and MinP) to that of computing *any* witness (as in NPMV), or non-witnesses (as in coNPMV). First, note that trivially $\mathrm{NPSV} \subseteq \mathrm{MaxP} \cap \mathrm{MinP}$, since *the* output of an NPSV function is both the minimum and the maximum. Similarly, $\mathrm{NPMV} \subseteq_c \mathrm{MaxP} \cap \mathrm{MinP}$. The more interesting question is whether these inclusions are strict. This is quite unlikely.

4.7 Theorem. $\mathrm{Max\,P} \subseteq \mathrm{NPMV}$ *if and only if* $\mathrm{Min\,P} \subseteq \mathrm{NPMV}$ *if and only if* $\mathrm{NP} = \mathrm{coNP}$.

Proof. If $\mathrm{NP} = \mathrm{coNP}$, then $\mathrm{NPMV} = \mathrm{F}\Delta_2^{\mathrm{P}}$ [Sel94], thus especially $\mathrm{NPMV} = \mathrm{Max\,P} = \mathrm{Min\,P}$.

Now suppose $\mathrm{Max\,P} \subseteq \mathrm{NPMV}$ (the case for Min P is analogous). Let A be any set from $\Delta_2^{\mathrm{P}} = \mathrm{P}^{\mathrm{NP}}$. In their paper on complexity classes of optimization problems, Vollmer and Wagner [VW95, Corollary 5.10] showed that there exists a function $f \in \mathrm{Max\,P}$ such that the least significant bit of f is equal to the characteristic function c_A of A. Thus, for all x, $x \in A$ if and only if $f(x)$ is odd. By assumption, f is in NPMV. Let this be witnessed by machine M. Then the following NP algorithm for A is obvious: Follow M's paths. If M prints an odd value, then accept. □

It is tempting to conclude from the proof of Theorem 3.7 that $\mathrm{Max\,P} \subseteq \mathrm{coNPMV}$. However, this too is unlikely.

4.8 Corollary. $\mathrm{Max\,P} \subseteq \mathrm{coNPMV} \iff \mathrm{Min\,P} \subseteq \mathrm{coNPMV} \iff \mathrm{NP} = \mathrm{coNP}$.

Proof. If $\mathrm{Max\,P} \subseteq \mathrm{coNPMV}$, then $\mathrm{NPMV} \subseteq_c \mathrm{Max\,P} \subseteq \mathrm{coNPMV}$, and therefore $\mathrm{NPMV} \subseteq_c \mathrm{coNPMV}$. But by Theorem 3.5, this implies $\mathrm{NP} = \mathrm{coNP}$. Conversely, if $\mathrm{NP} = \mathrm{coNP}$, then $\mathrm{NPMV}^{\mathrm{NP}} = \mathrm{NPMV}^{\mathrm{NP}\cap\mathrm{coNP}} = \mathrm{NPMV}$. This implies $\mathrm{MaxP} \subseteq \mathrm{NPMV}$, and since the hypothesis also implies $\mathrm{NPMV} = \mathrm{coNPMV}$, that $\mathrm{MaxP} \subseteq \mathrm{coNPMV}$.

□

The previous two results relativize. Thus it is easy to see that analogous results hold for higher levels of the NPMV hierarchy and Krentel's min/max hierarchy [FHOS93, VW95].

5. Relationships Between the Collapse of the Functional Difference and Polynomial Time Hierarchies

Chang and Kadin [CK90] showed that if the Boolean hierarchy over NP collapses to the k^{th} level, then the polynomial hierarchy collapses to the k^{th} level of the Boolean hierarchy over $\mathrm{NP}^{\mathrm{NP}}$. It is a simple consequence of known results that a similar connection exists for the functional hierarchy.

5.1 Theorem. *For any $k \geq 1$, if $\mathrm{NPMV}(k+1) = \mathrm{NPMV}(k)$ then $\Sigma\mathrm{MV}_3 = \mathrm{NPMV}^{\mathrm{NP}}(k)$.*

Proof. In the case of NP, the Boolean and difference hierarchies are the same. Thus, Chang and Kadin's theorem states that if $\mathrm{NP}(k+1) = \mathrm{NP}(k)$, then $\mathrm{PH} = \mathrm{NP}^{\mathrm{NP}}(k)$. Now we use two results of Fenner *et al.* [FHOS93] By Theorem 13 of their paper, the difference hierarchies over NPMV and NP are directly coupled: $\mathrm{NPMV}(k+1) = \mathrm{NPMV}(k)$ if and only if $\mathrm{NP}(k+1) = \mathrm{NP}(k)$. Note that the proof of the latter relativizes. Finally, from their Theorem 12, $\Sigma^p_{k+1} = \Sigma^p_k$ if and only if $\Sigma\mathrm{MV}_{k+1} = \Sigma\mathrm{MV}_k$. Thus,

$$\begin{aligned}
\mathrm{NPMV}(k+1) = \mathrm{NPMV}(k) &\Rightarrow \mathrm{NP}(k+1) = \mathrm{NP}(k) \\
&\Rightarrow \Sigma^p_3 = \mathrm{NP}^{\mathrm{NP}}(k) \\
&\Rightarrow \Sigma\mathrm{MV}_3 = \mathrm{NPMV}^{\mathrm{NP}}(k).
\end{aligned}$$

□

Since $\mathrm{NP}^A(k) \subseteq \mathrm{P}^{\mathrm{NP}^A[k]}$ for any oracle A, Chang and Kadin's theorem has as a corollary that if $\mathrm{NP}(k+1) = \mathrm{NP}(k)$, then $\Sigma^p_3 = \mathrm{P}^{\mathrm{NP}^{\mathrm{NP}}[k]}$ (indeed, they prove this directly in their paper before treating the stronger result). The functional analogue of such a collapse would be $\Sigma\mathrm{MV}_3 = \mathrm{FP}^{\mathrm{NPMV}^{\mathrm{NP}}[k]}$ or, equivalently, $\Sigma\mathrm{MV}_3 = \mathrm{FP}^{\mathrm{coNPMV}[k]}$. We cannot expect this as a direct consequence of Theorem 5.1, since the difference and query hierarchies are not intertwined in this context. Nevertheless, such an analogous result does hold, by the *proof* of the Chang and Kadin theorem.

5.2 Theorem. *If $\mathrm{NPMV}(k+1) = \mathrm{NPMV}(k)$ then $\Sigma\mathrm{MV}_3 = \mathrm{NPMV} \circ \mathrm{FP}^{\mathrm{coNPMV}[k-1]}$.*

Proof. In the appendix. □

Acknowledgment

We appreciate the contributions of J. Ramachandran to the results presented here.

References

[BLS84] R. Book and T. Long and A. Selman Quantitative relativizations of complexity classes *SIAM J. on Comput.*, 13(3):461–487, August 1984.

[CK90] R. Chang and J. Kadin. The boolean hierarchy and the polynomial hierarchy: a closer connection. In *Proceedings of the 5th Annual Conf. on Structure in Complexity Theory*, pages 140–153, New York, 1990. IEEE Computer Society Press.

[FHOS93] S. Fenner, S. Homer, M. Ogihara, and A. Selman. On using oracles that compute values. In *Proc. 10th Annual Symp. on Theoret. Aspects of Computer Science, Lecture Notes in Computer Science*, volume 665, pages 398–407. Springer-Verlag, 1993.

[Kad88] J. Kadin. The polynomial time hierarchy collapses if the Boolean hierarchy collapses. *SIAM J. on Comput.*, 17(6):1263–1282, December 1988.

[Kre88] M. Krentel. The complexity of optimization problems. *J. Comput. System Sci.*, 36:490–509, 1988.

[Kre92] M. Krentel. Generalizations of OptP to the polynomial hierarchy. *Theor. Com. Sci.*, 97:183–198, 1992.

[Pap84] C. Papadimitriou. On the complexity of unique solutions. *J. of the ACM*, 31:392–400, 1984.

[Ram94] J. Ramachandran. The polynomial time function hierarchy. Technical Report no. 94-50, Department of Computer and Information Sciences, Ohio State Univeristy, 1994.

[Sel94] A. Selman. A taxonomy of complexity classes of functions. *J. Comput. System Sci.*, 48(2):357–381, 1994.

[VW93] H. Vollmer and K. Wagner. The complexity of finding middle elements. *International Journal of Foundations of Computer Science*, 4:293–307, 1993.

[VW95] H. Vollmer and K. Wagner. Complexity classes of optimization functions. *Information and Computation*, 120:198–219, 1995.

[Wag87] K. Wagner. More complicated questions about maxima and minima, and some closures of NP. *Theoret. Comput. Sci.*, 51:53–80, 1987.

[Wag90] K. Wagner. Bounded query classes. *SIAM J. Comput.*, 19:833–846, 1990.

A. Technical Appendix: Proof of Theorem 5.2

Since it is equivalent, we take as our hypothesis, $\mathrm{NP}(k+1) = \mathrm{NP}(k)$. This clearly implies $\mathrm{NP}(k) = \mathrm{coNP}(k)$.

In order to explain how Chang and Kadin's proof gives this result, we recall some of their definitions, with some minor modifications in notation (for greater detail, we refer the reader to their paper [CK90]). Denote the $\leq_m^p$-complete language for $\mathrm{NP}(k)$ (respectively $\mathrm{coNP}(k)$) as $L_{\mathrm{NP}(k)}$ (respectively $L_{\mathrm{coNP}(k)}$). For example, $L_{\mathrm{NP}(1)} = \mathrm{SAT}$ and $L_{\mathrm{NP}(2)} = \{\langle x_1, x_2\rangle | x_1 \in \mathrm{SAT}$ and $x_2 \in \overline{\mathrm{SAT}}$. Since, by hypothesis, $\mathrm{NP}(k) = \mathrm{coNP}(k)$, it follows that $L_{\mathrm{NP}(k)} \leq_m^p L_{\mathrm{coNP}(k)}$. The basic idea underlying the Chang and Kadin proof is that such a reduction induces a reduction from an initial segment of SAT to an initial segment of $\overline{\mathrm{SAT}}$. This is done inductively via the notion of a "hard sequence," which is a j-tuple that, together with a $\leq_m^p$-reduction from $\mathrm{NP}(k)$ to $\mathrm{coNP}(k)$, can be used to find a $\leq_m^p$-reduction from $\mathrm{NP}(k-j)$ to $\mathrm{coNP}(k-j)$.

A.1 Definition. Let $L_{\mathrm{NP}(k)} \leq_m^p L_{\mathrm{coNP}(k)}$ via some polynomial time function h. Then we call $\langle 1^m, x_1, ..., x_j\rangle$ a *hard sequence* with respect to h for length m of order j, if $j = 0$ or if the following hold:

1. $1 \leq j \leq k-1$.
2. $|x_j| \leq m$.
3. $x_j \in \overline{\mathrm{SAT}}$.
4. $\langle 1^m, x_1, ..., x_{j-1}\rangle$ is a hard sequence with respect to h.
5. For all $y_1, ..., y_\ell \in \Sigma^{\leq m}$ (where $\ell = k - j$),

$$\pi_{\ell+1} \circ h(\langle y_1, ..., y_\ell, x_j, ..., x_1\rangle) \in \overline{\mathrm{SAT}}.$$

A hard sequence is called *maximal* if it cannot be extended to a hard sequence of a higher order. In this case the order of the sequence j is said to be maximal.

We can now outline the proof. Chang and Kadin's Lemma 3 [CK90] states that, given a maximal hard sequence for an appropriate (polynomially bounded) length, an NP machine can recognize an initial segment of the canonical complete language for $\mathrm{NP}^{\mathrm{NP}}$. That is, with the aid of such a sequence we can replace a Σ_2^p machine with an NP machine. Thus it suffices to find a maximal hard sequence to collapse the NP's of a $\Sigma\mathrm{MV}_3 = \mathrm{NPMV}^{\Sigma_2^p}$ machine.

Our principle observation is this: *Hard sequences of any given order can be obtained by a single query to a coNPMV oracle.* This can easily be seen as follows. Define the function $H : 1^+ \times \mathbf{N} \to \Sigma^*$ such that $H(1^m, j) \mapsto \langle 1^m, x_1, ..., x_j\rangle$ if and only if $\langle 1^m, x_1, ..., x_j\rangle$ is a hard sequence for length m of order j. It follows from Definition A.1 that the set of hard sequences is in coNP [CK90]; hence $graph(H) \in \mathrm{coNP}$, so that $H \in \mathrm{coNPMV}$. Therefore, we can obtain a maximal hard sequence for the appropriate polynomial length $m = p(|x|)$ by querying a coNPMV oracle for the value of $H(m, j)$ for j varying from 1 to $k-1$. We then feed the resulting maximal hard sequence, along with the original input x, to an NPMV machine that can, via the induced reduction from coNP to NP, collapse the NP oracles in an $\mathrm{NPMV}^{\mathrm{NP}^{\mathrm{NP}}}$ computation.