Abstract. This paper focuses on complexity classes of partial functions that are computed in polynomial time with oracles in NPMV, the class of all multivalued partial functions that are computable nondeterministically in polynomial time. Concerning deterministic polynomial-time reducibilities, it is shown that

1. A multivalued partial function is polynomial-time computable with $k$ adaptive queries to NPMV if and only if it is polynomial-time computable via $2^k - 1$ nonadaptive queries to NPMV.

2. A characteristic function is polynomial-time computable with $k$ adaptive queries to NPMV if and only if it is polynomial-time computable with $k$ adaptive queries to NP.

3. Unless the Boolean hierarchy collapses, $k$ adaptive (nonadaptive) queries to NPMV is different than $k+1$ adaptive (nonadaptive) queries to NPMV for every $k$.

Nondeterministic reducibilities, lowness and the difference hierarchy over NPMV are also studied. The difference hierarchy for partial functions does not collapse unless the Boolean hierarchy collapses, but, surprisingly, the levels of the difference and bounded query hierarchies do not interleave (as is the case for sets) unless the polynomial hierarchy collapses.

1 Introduction

In this paper we study classes of partial functions that can be computed in polynomial time with oracles in NPMV and NPSV; namely, we study the classes $\text{PF}^{\text{NPMV}}$ and $\text{PF}^{\text{NPSV}}$.

NPMV is the set of all partial multivalued functions that are computable nondeterministically in polynomial time, and NPSV is the set of all partial functions.
in this class that are single-valued. NPMV captures the complexity of computing witnesses to problems in NP. For example, let $sat$ denote the partial multivalued function defined by $sat(x)$ maps to a value $y$ if and only if $x$ encodes a formula of propositional logic and $y$ encodes a satisfying assignment of $x$. Then, $sat$ belongs to NPMV, and the domain of $sat$ (i.e., the set of all words $x$ for which the output of $sat(x)$ is non-empty) is the NP-complete satisfiability problem, SAT. Also, NPMV captures the complexity of inverting polynomial time honest functions. To wit, the inverse of every polynomial time honest function belongs to NPMV, and the inverse of every one-one polynomial time honest function belongs to NPSV.

The class of partial functions with oracles in NP, namely, PF$^{\text{NP}}$ has been well-studied [Kre88, Bei88], as have been the corresponding class of partial functions that can be computed nonadaptively with oracles in NP, viz. PF$^{\text{NP} \text{it}}$ [Sel92], and the classes of partial functions that are obtained by limiting the number of queries to some value $k \geq 1$, namely, PF$^{\text{NP}[k]}$ and PF$^{\text{NP}[k] \text{it}}$ [Bei91]. A rich body of results is known about these classes.

Here we raise the question, "What is the difference between computing with an oracle in NPMV versus an oracle in NP?" The answer is not obvious. If the partial function $sat$ is provided as an oracle to some polynomial-time computation $M$, then on a query $x$, where $x$ encodes a satisfiable formula of propositional logic, the oracle will return some satisfying assignment $y$. However, if the oracle to $M$ is the NP-complete set SAT, then to this query $x$, the oracle will only return a Boolean value "yes." On the other hand, by the well-known self-reducibility of SAT, $M$ could compute $y$ for itself by judicious application of a series of adaptive queries to SAT. Indeed Theorem 2 states that unbounded access to an oracle in NPMV is no more powerful than such an access to an oracle in NP. However, in Section 3 we will see that the situation for bounded query classes is much more subtle. In general, function oracles cannot be replaced by set oracles—but set oracles are still useful. We will show that every partial multivalued function in PF$^{\text{NP}[k]}$ can be computed by a partial multivalued function of the form $f \circ g$, where $f$ is in NPMV and $g$ is a single-valued function belonging to PF$^{\text{NP}[k]}$. Moreover, most surprisingly, the relationship between access to an oracle in NPMV and access to an oracle in NP is tight regarding set recognition; that is, $p^{\text{NP}[k]} = p^{\text{NP}[k]}$. This means that when we are computing characteristic functions, $k$ bounded queries to an oracle in NPMV give no more information than the same number of queries to an oracle in NP.

We will show that the levels of the nonadaptive and adaptive bounded query hierarchies interleave (for example, $k$ adaptive queries to a partial function in NPMV is equivalent to $2^k - 1$ nonadaptive queries to a partial function in NPMV), and we will show that these bounded query hierarchies collapse only if the Boolean hierarchy collapses.

In Section 4 we study nondeterministic polynomial time reductions to partial functions in NPMV. Unlike the case for deterministic functions, we will see that just one query to an NP oracle can substitute for an unbounded number of queries to any partial function in NPMV. The hierarchy that is formed by iteratively applying NP reductions is an analogue of the polynomial hierarchy, and we will show that this hierarchy collapses if and only if the polynomial hierarchy collapses.