On Functions Computable with Nonadaptive Queries to NP

Harry Buhrman* Jim Kadin† Thomas Thierauf‡

Abstract

We study $\mathrm{FP}_{\|}^{\mathrm{NP}}$*, the class of functions that can be computed with nonadaptive queries to an NP oracle. We show that optimization problems stemming from the known NP complete sets, where the optimum is taken over a polynomially bounded range, are hard for* $\mathrm{FP}_{\|}^{\mathrm{NP}}$*. This is related to (and, in some sense, extends) work of Chen and Toda [CT91]. In addition, it turns out that these optimization problems are all equivalent under a certain functional reducibility.*

By studying the question whether these function classes are complete for $\mathrm{FP}_{\|}^{\mathrm{NP}}$*, i.e., whether it is possible to compute an optimal value for a given optimization problem in* $\mathrm{FP}_{\|}^{\mathrm{NP}}$*, we show that this is exactly as hard as to compute membership proofs for NP complete sets in* $\mathrm{FP}_{\|}^{\mathrm{NP}}$*. On the other hand,* $\mathrm{FP}_{\|}^{\mathrm{NP}}$ *can be* characterized *as the class of functions that reduces to the above mentioned optimization functions. We will call this property* quasi-completeness.

A subclass of $\mathrm{FP}_{\|}^{\mathrm{NP}}$ *is NPSV, the class of functions that can be computed by single-valued NP transducers. We exhibit function classes that are quasi-complete for NPSV but not complete unless the Polynomial Time Hierarchy collapses.*

1 Introduction

A fundamental issue in the study of NP and related classes is the complexity of generating proofs that a string is a member of a given language.[1] For NP complete sets, it is well known that the lexicographically least, or leftmost, witness of membership can be generated in $\mathrm{FP}^{\mathrm{NP}}$, the class of functions computable in polynomial time with access to an oracle in NP [Va76]. For example, if we consider the NP complete set SAT, the following function is in $\mathrm{FP}^{\mathrm{NP}}$.

$$f_{left}(\varphi) = \begin{cases} \text{the leftmost satisfying} & \\ \text{assignment of } \varphi, & \text{if } \varphi \in \mathrm{SAT}, \\ \perp, & \text{if } \varphi \notin \mathrm{SAT}, \end{cases}$$

where $\perp$ is some special symbol to denote that $f(x)$ is undefined.

Krentel showed that every function in $\mathrm{FP}^{\mathrm{NP}}$ can be reduced to f_{left}, and thus f_{left} is complete for $\mathrm{FP}^{\mathrm{NP}}$ [Kr86]. His proof involved showing that the leftmost accepting path of an NP computation can be made to correspond to the correct query path of a $\mathrm{FP}^{\mathrm{NP}}$ computation and to the leftmost satisfying assignment in the output of Cook's reduction to SAT [Co71].

In this paper, we consider $\mathrm{FP}_{\|}^{\mathrm{NP}}$, the class of functions in $\mathrm{FP}^{\mathrm{NP}}$ that can be computed by making nonadaptive queries to NP, that is, all the queries must be written down before any answers are received from the oracle. We are also interested in $\mathrm{P}_{\|}^{\mathrm{NP}}$, the class of languages recognizable in polynomial time by making nonadaptive queries to NP. See for an extensive study [Ka88].

More specifically, our work is motivated by the following questions.

(1) What is the structure of $\mathrm{FP}_{\|}^{\mathrm{NP}}$? What functions are in $\mathrm{FP}_{\|}^{\mathrm{NP}}$? What functions are hard (or complete) for $\mathrm{FP}_{\|}^{\mathrm{NP}}$?

(2) Can proofs of membership for NP complete sets (for example satisfying assignments) be computed by functions in $\mathrm{FP}_{\|}^{\mathrm{NP}}$?

(3) Are there classes of proofs that are easier to generate than the leftmost proof? What is the relationship between different proofs? Do all proofs

*Dept. Llenguatges i Sist. Informàtics Univ. Politècnica de Catalunya Pau Gargallo 5, 08028 Barcelona, Spain. E-mail: harry@goliat.upc.es. Supported by a TALENT stipendium from the Netherlands Organization for Scientific Research (NWO) and ESPRIT Basic Research Actions Program of the EC (under contract No. 7141 (project ALCOM II). Part of this research was done while visiting Boston University with the support of NSF Grant CCR-8814339

†Department of Computer Science, University of Maine, 5752 Neville Hall, Orono, Maine 04469-5752.

‡Abteilung Theoretische Informatik, Universität Ulm, Oberer Eselsberg, 89069 Ulm, Germany. Part of the work done while visiting the Department of Computer Science at the University of Rochester, Rochester, NY and the Department of Computer Science at the University of Maine, Orono. Supported in part by DFG Postdoctorial Stipend Th 472/1-1 and NSF grant CCR-8957604.

[1] The proof generating functions that we discuss in this paper are called *inverse functions* in [WT90].

of membership contain the same amount of information?

With respect to (1), Chen and Toda [CT93] showed many functions to be complete for $\mathrm{FP}_{\|}^{\mathrm{NP}}$ (see also [JT93]). An example is the function that for any Boolean formula φ gives the supremum of the satisfying assignments of φ. Observe that the supremum must not necessarily be a satisfying assignment of φ.

In another paper, Chen and Toda [CT91] showed that under certain restrictions, every function in $\mathrm{FP}_{\|}^{\mathrm{NP}}$ can be reduced to any function that computes optimal solutions to NP complete optimization problems. They define an NPCOP to be an NP optimization problem for which the cost of the solutions is bounded by a polynomial in the size of the problem instance. For example, the size of the largest clique in a graph is bounded by the size of the graph. They show that NPCOP's that are linearly paddable (see [CT91] for precise definitions), are hard for $\mathrm{FP}_{\|}^{\mathrm{NP}}$. We extend their work to a, in some sense, broader class of functions.

In an NPCOP, the set of instances for which one wants to compute an optimal solution has to be a polynomial-time decidable set. We will define the class of *polynomially bounded NP optimization problems*, NPbOpt for short, where the set of instances can be an NP set. The advantage is that now *any* NP set together with some polynomially bounded solution cost function defines an NPbOpt. We will show many NPbOpt's to be hard for $\mathrm{FP}_{\|}^{\mathrm{NP}}$. For this, we consider NP sets that have universal relations [AB92]. Sets that have universal relations are NP complete sets that have related proof systems, i.e., it is possible to compute from a proof of membership for an instance for one set a proof of membership for an instance for the other. We will show in Section 3 that every universal NPbOpt having an embedding operator (see next section for definitions) is hard for $\mathrm{FP}_{\|}^{\mathrm{NP}}$. An example for such a function class is F_{zero}.

$$f \in F_{zero} \iff$$

$$f(\varphi) = \begin{cases} \text{some satisfying assignment} & \\ \text{of } \varphi \text{ having the maximum} & \\ \text{number of zeros,} & \text{if } \varphi \in \mathrm{SAT}, \\ \perp, & \text{if } \varphi \notin \mathrm{SAT}. \end{cases}$$

Moreover, we will show that $\mathrm{FP}_{\|}^{\mathrm{NP}}$ is *precisely* the class of functions that reduces to universal NPbOpt's having embedding operators. We will call this property *quasi-completeness*. Furthermore we will show that functions computing solutions to universal NPbOpt's having embedding operators are in fact *equivalent* under an appropriate functional reducibility.

It thus follows that $\mathrm{FP}_{\|}^{\mathrm{NP}}$ is exactly the class of functions that reduce to F_{zero} and hence that F_{zero} is quasi-complete for $\mathrm{FP}_{\|}^{\mathrm{NP}}$.

In Section 4, we link the question whether these functions are complete in the classical sense – *some member* has to be computable in $\mathrm{FP}_{\|}^{\mathrm{NP}}$ – to the complexity of generating proofs of membership for certain NP complete sets. For example consider the following function class.

$$f \in F_{sat} \iff$$

$$f(\varphi) = \begin{cases} \text{some satisfying} & \\ \text{assignment of } \varphi, & \text{if } \varphi \in \mathrm{SAT}, \\ \perp, & \text{if } \varphi \notin \mathrm{SAT}. \end{cases}$$

We show that some function in F_{zero} is computable in $\mathrm{FP}_{\|}^{\mathrm{NP}}$ if and only if some function in F_{sat} is computable in $\mathrm{FP}_{\|}^{\mathrm{NP}}$. In other words, if it is at all possible to compute some satisfying assignment within $\mathrm{FP}_{\|}^{\mathrm{NP}}$, then this can be done for F_{zero}.

NPSV is the class of functions that can be computed by single-valued NP transducers. NPSV is a subclass of $\mathrm{FP}_{\|}^{\mathrm{NP}}$. Is it possible to compute satisfying assignments even within NPSV? Hemaspaandra et.al. [HNOS93] give a negative answer: this is *not* possible within NPSV, unless the Polynomial Time Hierarchy collapses to the second level, i.e., $F_{sat} \bigcap \mathrm{NPSV} \neq \emptyset \implies \mathrm{PH} = \Sigma_2^P$.

An improvement of their result to $\mathrm{FP}_{\|}^{\mathrm{NP}}$ will therefore give a negative answer to the completeness (in the classical sense) of functions computing solutions to NPbOpt's having embedding operators. In Section 5, we will will make a first step by extending their result to $\mathrm{FP}^{\mathrm{NPSV}[1]}$. Furthermore, we show that F_{sat} is quasi-complete for NPSV, $\mathrm{FP}^{\mathrm{NPSV}[1]}$, and $\mathrm{FP}_{\|}^{\mathrm{NPSV}}$ under certain reducibilities, respectively. As a consequence, we can improve a result of Watanabe and Toda [WT90]. They show that f_{left} cannot be reduced to F_{sat}, unless $\mathrm{NP} \neq \text{co-NP}$. We show that even $\mathrm{FP}^{\mathrm{NPSV}[1]}$ cannot be reduced to F_{sat}, unless $\mathrm{NP} \neq \text{co-NP}$. (Note that f_{left} is hard for $\mathrm{FP}^{\mathrm{NPSV}[1]}$.)

The following is a summary of our main results:

(1) We extend the work of [CT91] and show that a broad class of functions, that are interreducible, is hard for $\mathrm{FP}_{\|}^{\mathrm{NP}}$ (Theorem 3.1 and 3.2).

(2) In an attempt to show that these hardness results cannot be extended to completeness results we provide a link to the problem of generating proofs of membership for certain NP complete sets (Theorem 4.2).

(3) We will characterize $\mathrm{FP}_{\|}^{\mathrm{NP}}$ as the class of functions that is (metric) reducible to F_{zero} (Theorem 4.3), and NPSV, $\mathrm{FP}_{\|}^{\mathrm{NPSV}}$, and $\mathrm{FP}_{\|}^{\mathrm{NP}}$ as those functions that are reducible to F_{sat} for certain functional reducibilities, respectively (Theorem 5.4). We will call this property quasi-completeness.

(4) As a consequence of the quasi-completeness results, we strengthen a result of Watanabe and Toda [WT90] (Corollary 5.5).

2 Preliminaries

2.1 Universal Relations

Let $\Sigma = \{0, 1\}$ be an alphabet and let R be a relation on $\Sigma^* \times \Sigma^*$. The *domain of* R is the set $D_R = \{ x \in \Sigma^* \mid \exists y \in \Sigma^* \; xRy \}$. Any $y \in \Sigma^*$ witnessing that some x is in D_R is called a *solution for x with respect to* R. The set of all solutions for some $x \in \Sigma^*$ with respect to R is denoted by $S_R(x)$, and S_R is the set of all solutions.

We say that R is a *NP relation*, if the following two conditions hold.

(i) There is a polynomial p such that for all $x \in D_R$, any solution for x has length $p(|x|)$, i.e., $S_R(x) \subseteq \Sigma^{p(|x|)}$, and

(ii) R is decidable in polynomial time.

By F_R, we denote the class of all functions $f : \Sigma^* \to S_R \cup \{\perp\}$ such that for all $x \in \Sigma^*$

$$f(x) = \begin{cases} \text{some } y \in S_R, & \text{if } x \in D_R, \\ \perp, & \text{otherwise.} \end{cases}$$

In this paper, we investigate how hard it is to compute some solution for a given x with respect to some NP relation, i.e., some function in F_R. For comparing the complexity of this task for different NP relations, we need to consider reductions between them. Here, the following question arises. Given two NP relations R_0 and R_1 such that we can many-one reduce D_{R_0} to D_{R_1} via some FP function h. For any $x \in D_{R_0}$, can we compute a solution for x with respect to R_0 from a given solution for $h(x)$ with respect to R_1? In general, this is not known. But in some cases, we indeed can. Consider for example Cook's reduction from an arbitrary NP set to SAT [Co71]. For a fixed NP machine M, for any input x for M there is constructed a Boolean formula φ_x such that x is accepted by M iff $\varphi_x \in$ SAT. And in fact, from any satisfying assignment for φ_x, one can easily compute an accepting path of M on input x. The same holds for the many-one reductions for example from SAT to the known NP complete sets such as HAM or CLIQUE which one finds in the standard textbooks (see for example in [HU79]).

But in general, a many-one reduction doesn't respect the structure of the solution spaces of the instances that are mapped to each other. It just guarantees the existence/nonexistence of solutions. However, in a very general approach, Agrawal and Biswas [AB92] showed that a large variety of NP complete sets in fact share the above described property with SAT, including all sets being isomorphic to SAT. They introduced the notion of an *universal relation* which we define below.

Let $x = x_1 \cdots x_n$ be a string of length n and $\alpha = (n_1, \ldots, n_l)$ a sequence of integers, where $1 \leq n_i < n_{i+1} \leq n$, for $i = 1, \ldots, l-1$. Then $Mask(x, \alpha)$ is the string of length l, where the i-th bit is the n_i-th bit of x, i.e., $Mask(x, \alpha) = x_{n_1} \cdots x_{n_l}$. For a set $S \subseteq \Sigma^n$, $Mask(S, \alpha) = \{ Mask(x, \alpha) \mid x \in S \}$.

An NP relation R has a *join operator*, if there are two FP functions $join_R$ and $join\text{-}list_R$ such that if $join_R(x_1, \ldots, x_n) = z$ and $join\text{-}list_R(x_1, \ldots, x_n) = \alpha$, where $x_1, \ldots, x_n, z \in \Sigma^*$ and α is a sequence of integers, then

$$Mask(S_R(z), \alpha) =$$
$$\{ y_1 \cdots y_n \mid y_i \in S_R(x_i), \text{ for } i = 1, \ldots, n \}.$$

That is, the join function combines several given strings into one string z in such a way that from any solution for z, we can compute solutions for the given strings: we just have to read the bit positions to which the *join-list* function is pointing to.

An NP relation R has an *equivalence operator*, if there are two FP functions equ_R and $equ\text{-}list_R$ such that if $equ_R(x, (i_1, \ldots, i_k), (j_1, \ldots, j_k)) = z$ and $equ\text{-}list_R(x, (i_1, \ldots, i_k), (j_1, \ldots, j_k)) = \alpha$, where $x \in \Sigma^*$, $k \leq p(|x|)$, and $1 \leq i_l \neq j_l \leq p(|x|)$ for $l = 1, \ldots, k$, where polynomial p gives the length of the solutions of R, then

$$Mask(S_R(z), \alpha) =$$
$$\{ y \in S_R(x) \mid \text{the } i_l\text{-th and the } j_l\text{-th bit}$$
$$\text{of } y \text{ are equal, for } l = 1, \ldots, k. \}$$

That is, the equivalence function maps a given string x to some string z in such a way that the solution space of z at the positions to which the equivalence-list function is pointing to, consists of all the solutions of x

where the bit positions indicated by the two input integer sequences are equal.

For getting the corresponding decision problem D_R of an NP relation R NP complete, besides having join and equivalence operators, we need an instance $b \in \Sigma^*$ such that the solution space of b has a certain structure. Namely, there has to exist three integers $\alpha = (n_1, n_2, n_3)$ such that $Mask(S_R(b), \alpha) = \Sigma^3 - \{000\}$. Such a string b is called a *building block*. A building block ensures that we are able to exclude one of the eight potential solutions patterns b can have at three positions.[2]

An NP relation having a join and an equivalence operator and a building block is called *universal*. An NP set L is universal, if there is an universal NP relation R such that $L = D_R$.

For example SAT is universal. The join function is essentially the conjunction. That is, for any two Boolean formulas φ_1 and φ_2, after renaming the variables so that φ_1 and φ_2 have disjoint sets of variables, $join_{\mathrm{SAT}}(\varphi_1, \varphi_2) = \varphi_1 \wedge \varphi_2$. For the equivalence operator, let φ be a Boolean formula and x and y be two variables occurring in φ. Then the formula $\varphi \wedge (x \leftrightarrow y)$ has as satisfying assignments all satisfying assignments of φ where x and y have the same value. $b = \overline{x} \vee \overline{y} \vee \overline{z}$ is a building block for SAT. Moreover, Agrawal and Biswas [AB92] showed that all sets that are isomorphic to SAT have an universal relation.

Theorem 2.1. **[AB92]** Let R_1 be an universal relation. Then the following holds.

(i) D_{R_1} is NP complete.

(ii) For any NP relation R_0 there are two FP functions h and *h-list* such that for any $x \in \Sigma^*$, we have $Mask(S_{R_1}(h(x)), h\text{-}list(x)) = S_{R_0}(x)$.

Part (ii) of this theorem answers our above question for the class of universal sets. Namely, h is a many-one reduction from D_{R_0} to D_{R_1} and for any $x \in D_{R_0}$, via function *h-list*, we can compute a solution for x with respect to R_0 from a given solution for $h(x)$ with respect to R_1.

2.2 Polynomially Bounded NP Optimization Problems (NPbOpt)

Let R be a relation on $\Sigma^* \times \Sigma^*$. A FP function $c : D_R \times S_R \to \mathbf{N}$ is called a *solution cost function for R*. The *optimal solution cost function for R*, $c^* : D_R \to \mathbf{N}$ is defined by

$$c^*(x) = \max\{\, c(x,y) \mid y \in S_R(x) \,\}.$$

For any $x \in D_R$, we define the set of optimal solutions by $\mathrm{OptSol}_{R,c} = \{\, y \in S_R(x) \mid c(x,y) = c^*(x) \,\}$. For any NP relation R and any cost function c for R, we say that (R, c) is an *NP optimization problem*, namely the problem to compute an optimal solution for any given $x \in D_R$. For any function $f : \Sigma^* \to S_R \cup \{\perp\}$, we define

$$f \in \mathrm{Opt}_{R,c} \iff \forall x \in \Sigma^* :$$

$$f(x) = \begin{cases} \text{some } y \in \mathrm{OptSol}_{R,c}, & \text{if } x \in D_R, \\ \perp, & \text{otherwise.} \end{cases}$$

(R, c) is called a *polynomially bounded NP optimization problem*, NPbOpt for short, if there is a polynomial that bounds c. Note that from any NP set we can derive a polynomially bounded NP optimization problem by taking some relation witnessing the set being in NP and some arbitrary polynomially bounded cost function c. In contrast, Chen and Toda defined the more restricted notion of an NPCOP which is defined similar to an NPbOpt, but with the additional constraint that the domain D_R is a set in P.

(R, c) is called *universal*, if R is universal and the join operation of R respects c, i.e., if we join two instances, then we can compute an optimal solution for the two instances from an optimal solution for the join of the instances. More formally, there has to exist a FP function $g : D_R \times D_R \times S_R \to S_R \times S_R$ such that for all $x_0,\, x_1 \in D_R$ and for all $y \in \mathrm{OptSol}_{R,c}(join_R(x_0, x_1))$, if $g(x_0, x_1, y) = (y_0, y_1)$, then $y_0 \in \mathrm{OptSol}_{R,c}(x_0)$ and $y_1 \in \mathrm{OptSol}_{R,c}(x_1)$.

We say that (R, c) has an *embedding operator*, if there exist two FP functions $e : \Sigma^* \to D_R$ and $g : \Sigma^* \times S_R \to S_R$ such that there is a $f \in \mathrm{Opt}_{R,c}$ such that for all $x \in \Sigma^*$

$$\forall y \in \mathrm{OptSol}_{R,c}(e(x)) : \quad g(x, y) = f(x).$$

That is, e maps every string x to some string z in the domain D_R of R such that from an optimal solution for z, one can either compute an optimal solution for x, or detect that $x \notin D_R$.

2.3 Examples

First of all, note that all examples of Chen and Toda for NPCOP's can easily be modified to be universal NPbOpt's having an embedding operator. They

[2] The building block defined in [AB92] uses four positions and is a bit more complicated. But if a relation has a join and an equivalence operator, then our building block requirement is equivalent to that in [AB92].

mention for example Maximum Clique, Minimum Coloring, and Longest Path. We give some examples that might not be expressible as an NPCOP.

Since the number of zeros in an assignment of a Boolean formula is bounded by the number of variables of that formula, F_{zero} is an NPbOpt. Since SAT is universal and since the join, which is the conjunction here, respects the maximum number of zeros, F_{zero} is universal. Furthermore, F_{zero} has an embedding operator. Let $\varphi = \varphi(x_1, \ldots, x_n)$ be a Boolean formula and let $z_1, \ldots, z_{n+1}$ be new variables. Define

$$\Phi(x_1, \ldots, x_n, z_1, \ldots, z_{n+1}) = \varphi \vee (z_1 \wedge \cdots \wedge z_{n+1}).$$

Then $\Phi \in$ SAT, and if $\varphi \in$ SAT and a is a satisfying assignment with the maximum number of zeros for φ, then $a0^{n+1}$ is a satisfying assignment for Φ with the maximum number of zeros. On the other hand, if $\varphi \notin$ SAT, then $0^n 1^{n+1}$ is a satisfying assignment for Φ with the maximum number of zeros. Therefore, getting a satisfying assignment with the maximum number of zeros for Φ, one can either get one for φ or detect that $\varphi \notin$ SAT.

$F_{max0guess}$ is defined on instances $(N, x, 1^m)$ for the standard universal NP complete set, i.e., it is asked whether the nondeterministic Turing machine N accepts input x in at most m steps. Any nondeterministic computation path of N can be represented as a binary string corresponding to the nondeterministic branch points in the computation. $F_{max0guess}$ is the class of functions that give, on input $(N, x, 1^m)$, some accepting path of N on x with the maximum number of zeros, and are undefined, if there is no accepting path. Join, equivalence, and embedding operators are similar as for F_{zero}. Here, one has to manipulate the input machine N appropriately.

In unary-TSP, there is given an undirected graph G with integer weights given in unary notation on the edges, so that the weights are bounded by the size of the input. The task is to determine a traveling salesman tour in G having minimal weight.[3] For the join and equivalence operators see [AB92]. For the embedding operator, let B be the sum of the weights of the edges of G. Let G' be the extension of G to a complete graph, where all new edges have weight $B + 1$. Now, G' clearly has a traveling salesman tour. Furthermore, if the tour with minimum weight in G' is bounded by B, then this is also a minimum tour in G. Otherwise, there is no traveling salesman tour in G.

[3] Note that unary-TSP is a minimization problem. By defining an appropriate solution cost function, this can be easily turned into a maximization problem.

2.4 Functional Reducibilities

There are several notions of reducibility between functions. Krentel [Kr86] introduced the *metric reduction*. Let f, g be functions.

$$f \leq_{1\text{-}T}^{FP} g \iff \exists t_1, t_2 \in \mathrm{FP}: \quad f(x) = t_2(x, g \circ t_1(x)).$$

This clearly captures the idea of being able to compute $f(x)$ from one call to g.

Watanabe and Toda [WT90] and Chen and Toda [CT91] extended this reduction to function classes. Let G be a class of functions. We distinguish the case that *one pair* of translation functions reduces f to all functions in G, or that for each function $g \in G$ there is a pair of translation functions that reduces f to g. In the first case, we call the reduction *uniform*.

$$f \leq_{1\text{-}T}^{uniform\text{-}FP} G \iff$$

$$\exists t_1, t_2 \in \mathrm{FP}\ \forall g \in G: \quad f(x) = t_2(x, g \circ t_1(x)),$$

and

$$f \leq_{1\text{-}T}^{FP} G \iff$$

$$\forall g \in G\ \exists t_1, t_2 \in \mathrm{FP}: \quad f(x) = t_2(x, g \circ t_1(x)).$$

We also consider the more general type of reduction when more than one instance is given to a function in G. That is, $t_1(x)$ produces a list of instances and t_2 gets the function values of some function in G of these instances. This is called a *truth-table reduction* and denoted by $\leq_{tt}^{uniform\text{-}FP}$ and $\leq_{tt}^{FP}$, respectively.

If G is a class of partial functions, we must deal with the case that $f(x)$ is defined, while $g \circ t_1(x)$ is undefined. We call a reduction *strict*, denoted by $f \leq_{tt}^{FP\text{-}strict} G$, if there are FP functions t_1 and t_2 witnessing that $f \leq_{tt}^{FP} G$ such that for all x, if $f(x)$ is defined, then g is defined for all instances produced by $t_1(x)$.

Let $\leq_r$ be any of the reducibilities defined here. For a class F of functions, we say that G is hard for F with respect to $\leq_r$-reduction, if for all functions $f \in F$, we have $f \leq_r G$. This is denoted by $F \leq_r G$. Furthermore, we say that G is *complete for* F, if in addition there is some function in G that is also in F, i.e., $F \cap G \neq \emptyset$.

We also consider the case that eventually not all functions in F are reducible to G, but that any function in G can be used to compute *some* function in F. We call this a *weak* reduction.

$$F \leq_{1\text{-}T}^{weak\text{-}FP} G \iff$$

$$\forall g \in G\ \ \exists t_1, t_2 \in \mathrm{FP}: \quad t_2(x, g \circ t_1(x)) \in F.$$

The uniform and truth-table versions of this reduction are defined analogously. The *uniform weak Turing reduction* was defined in [FHOS93].

It is easy to see that all the reducibilities defined here are transitive, but in general, only the weak reducibilities are reflexive.

Although the uniformness condition seems to be a strong restriction on the reduction, Watanabe and Toda [WT90], using a proof technique from Grollmann and Selman [GS88], have shown that for many function classes these two reduction types are in fact equivalent.

Lemma 2.2. [**WT90**] Let f be a function, R an NP relation and c a solution cost function for R. Then we have

(i) $f \leq_{tt}^{FP} F_R \iff f \leq_{tt}^{uniform\text{-}FP} F_R$,

(ii) $f \leq_{tt}^{FP} \mathrm{Opt}_{R,c} \iff f \leq_{tt}^{uniform\text{-}FP} \mathrm{Opt}_{R,c}$.

The same holds for the other reducibilities defined above, i.e., $\leq_{1\text{-}T}^{FP}$, $\leq_{tt}^{FP\text{-}strict}$, $\leq_{1\text{-}T}^{FP\text{-}strict}$, $\leq_{1\text{-}T}^{weak\text{-}FP}$, and $\leq_{tt}^{weak\text{-}FP}$.

3 Function Classes Hard for $\mathrm{FP}_{\parallel}^{\mathrm{NP}}$

Chen and Toda [CT91] showed that certain NPCOP's are hard for $\mathrm{FP}_{\parallel}^{\mathrm{NP}}$ under $\leq_{1\text{-}T}^{FP}$-reductions. Our first theorem states that this holds as well for certain NPbOpt's, as for example F_{zero}.

Theorem 3.1. Let (R,c) be an universal NPbOpt having an embedding operator. Then we have $\mathrm{FP}_{\parallel}^{\mathrm{NP}} \leq_{1\text{-}T}^{FP} \mathrm{Opt}_{R,c}$.

Proof. Let $f \in \mathrm{FP}_{\parallel}^{\mathrm{NP}}$ via some polynomial-time transducer T and some NP set A. Let $x \in \Sigma^*$ be fixed. We show how to compute $f(x)$ when getting an arbitrary optimal solution for some instance z with respect to (R,c).

Let $w_1, \ldots, w_k$ be the queries of transducer T on input x to A. Since D_R is NP complete, there is a FP function h reducing A to D_R. Let e and g be embedding functions for (R,c) . We use e to map all strings $h(w_i)$ to D_R and then combine all the resulting strings into one string z using the join function of R. That is, we define

$$z = join_R(e \circ h(w_1), \ldots, e \circ h(w_k)).$$

Let $y \in \mathrm{OptSol}_{R,c}(z)$. Since $join_R$ respects c, from y we can compute solutions $y_i \in \mathrm{OptSol}_{R,c}(e \circ h(w_i))$, for $i = 1, \ldots, k$. Now, $g(h(w_i), y_i)$ either gives a witness that $h(w_i)$ is in D_R, and hence w_i is in A, or $g(h(w_i), y_i)$ is undefined, and hence w_i is not in A.

Thus, we can compute the answers to $w_1, \ldots, w_k$ from y, and therefore, we can compute $f(x)$. □

Our next theorem shows that any universal NPbOpt having an embedding operator is hard for any other NPbOpt under $\leq_{1\text{-}T}^{weak\text{-}FP}$-reductions, and hence, any such NPbOpt's are equivalent to each other.

Theorem 3.2. Let (R_0, c_0) be an universal NPbOpt having an embedding operator. Then we have $\mathrm{Opt}_{R,c} \leq_{1\text{-}T}^{weak\text{-}FP} \mathrm{Opt}_{R_0,c_0}$ for any NPbOpt (R,c).

Proof. We will show that for any $x \in \Sigma^*$, we can map x to some string $z \in D_{R_0}$ such that from an optimal solution for z with respect to (R_0, c_0), we can either compute an optimal solution for x with respect to (R,c) or detect that x is not in D_R.

Let us define the NP relation R' as follows. For any $x \in \Sigma^*$ and $k \leq p(|x|)$, where p is some polynomial that bounds the solution cost function c,

$$(x,k)R'y \iff xRy \text{ and } c(x,y) \geq k.$$

Let $x \in \Sigma^*$ be fixed and let k^* be the maximum k such that $(x,k) \in D_{R'}$. Observe that any solution for (x,k^*) with respect to R' is an optimal solution for x with respect to (R,c), i.e., $S_{R'}(x,k^*) \subseteq \mathrm{OptSol}_{R,c}(x)$. We will show how to compute a witness for each $(x,k) \in D_{R'}$ when getting an arbitrary optimal solution for some instance z with respect to (R_0,c_0). From these witnesses, we output the one for (x,k^*).

Since R' is an NP relation and since R_0 is universal, there is a FP function h that reduces $D_{R'}$ to D_{R_0} in such a way that for any $(x,k) \in D_{R'}$ and from any witness for $h(x,k) \in D_{R_0}$ we can compute a witness for $(x,k) \in D_{R'}$.

As in the proof of Theorem 3.1, using the embedding function and the join function for R_0, we combine all the resulting strings $h(x,1), \ldots, h(x,p(|x|))$ into one string z such that from a witness for $z \in D_{R_0}$, we can compute witnesses for all $h(x,k)$ that are in D_{R_0}. □

Corollary 3.3. Let (R,c) and (R',c') be universal NPbOpt's having an embedding operator. Then $\mathrm{Opt}_{R',c'} \equiv_{1\text{-}T}^{weak\text{-}FP} \mathrm{Opt}_{R,c}$

If we don't assume the existence of an embedding function for the NPbOpt's, then the above theorems still hold, but with the corresponding truth-table reductions, respectively.

4 Completeness

In the previous section, we established a framework for proving certain functions hard for $\mathrm{FP}_{\|}^{\mathrm{NP}}$. The natural question that arises is whether these functions are also complete for $\mathrm{FP}_{\|}^{\mathrm{NP}}$. (Recall that G is complete for F if G is hard for F and, in addition, $F \cap G \neq \emptyset$). Chen and Toda [CT91] showed that a *randomized version* of $\mathrm{FP}_{\|}^{\mathrm{NP}}$ can actually compute any NPCOP in the following sense: for any NPCOP there is a two-place function $f \in \mathrm{FP}_{\|}^{\mathrm{NP}}$, that, when given as one input the problem instance x and as the other input some randomly chosen string, outputs with high probability an optimal solution for x with respect to the given NPCOP. This result holds also for NPbOpt's.

Theorem 4.1. **[CT91]** Let (R, c) be an NPbOpt and let e be a polynomial. Then there exist a function $f \in \mathrm{FP}_{\|}^{\mathrm{NP}}$ and a polynomial r such that for all $x \in D_R$, $|x| = n$, we have

$$\mathrm{Prob}\{\, w \in \{0,1\}^{r(n)} \mid f(x,w) \in \mathrm{OptSol}_{R,c}(x)\,\}$$
$$\geq 1 - 2^{-e(n)}.$$

However, at present time, we do not know whether the NPbOpt results from the previous section can be extended to completeness results. Consider the following example concerning F_{zero}. Let $\varphi = \varphi(x_1, \ldots, x_n)$ be a Boolean formula and let $z_1, \ldots, z_n$ be new variables. Define

$$\Phi(x_1, \ldots, x_n, z_1, \ldots, z_n) = \varphi \wedge \bigwedge_{i=1}^{n} (x_i \leftrightarrow \overline{z_i}).$$

Then $\varphi \in \mathrm{SAT} \iff \Phi \in \mathrm{SAT}$, and every satisfying assignment for Φ has the same number of zeros, namely n. This already indicates a close connection between computing a satisfying assignment with the maximum number of zeros and computing an arbitrary satisfying assignment. And indeed, we can show that if it is at all possible to compute some satisfying assignment with parallel queries to NP, then this is also possible within F_{zero}. In other words, obtaining such a completeness result is exactly as hard as any proof that one can indeed compute some satisfying assignment in $\mathrm{FP}_{\|}^{\mathrm{NP}}$.

Theorem 4.2. Let (R, c) be an NPbOpt and R_0 be an universal relation. Then we have $F_{R_0} \cap \mathrm{FP}_{\|}^{\mathrm{NP}} \neq \emptyset \iff \mathrm{Opt}_{R,c} \cap \mathrm{FP}_{\|}^{\mathrm{NP}} \neq \emptyset$.

Proof. Let $f \in F_{R_0} \cap \mathrm{FP}_{\|}^{\mathrm{NP}}$. We define an NP relation R' as follows. For any $x \in \Sigma^*$ and $k \leq p(|x|)$, where p is some polynomial that bounds the solution cost function c,

$$(x,k) R' y \iff xRy \text{ and } c(x,y) \geq k.$$

Since R' is an NP relation, there are FP functions h and s such that h many-one reduces $D_{R'}$ to D_{R_0} and for any $(x,k) \in D_{R'}$ and any string z witnessing that $h(x,k) \in D_{R_0}$, $s(z)$ is a witness that (x,k) is in $D_{R'}$.

Let $x \in \Sigma^*$ be fixed. We show that we can compute some value in $\mathrm{OptSol}_{R,c}(x)$ with parallel queries to some NP set.

Let k^* be the maximal k such that $(x,k) \in D_{R'}$, i.e., we have $c^*(x) = k^*$. Then $h(x,k^*)$ is in D_{R_0} and $z = f \circ h(x,k^*)$ is some witness for this, by assumption. Hence, $s(z)$ is a witness that $(x,k^*) \in D_{R'}$ and therefore, we have that $s(z) \in \mathrm{OptSol}_{R,c}(x)$. That is, we define

$$f'(x) = s \circ f \circ h(x, c^*(x)).$$

It remains to show that $f' \in \mathrm{FP}_{\|}^{\mathrm{NP}}$. We leave this to the reader. □

On the other hand, we will show in the next theorem that all functions that are $\leq_{1\text{-}T}^{FP}$-reducible to some NPbOpt are already in $\mathrm{FP}_{\|}^{\mathrm{NP}}$, and therefore, together with Theorem 3.1, it follows that $\mathrm{FP}_{\|}^{\mathrm{NP}}$ can be *characterized* as the class of functions that are $\leq_{1\text{-}T}^{FP}$-reducible to some NPbOpt. This can be interpreted as a weaker form of completeness.

Definition Let F and G be function classes. We say that G is *quasi-complete* for F under $\leq_r$-reduction, if $F = \{\, f \mid f \leq_r G\,\}$.

The next theorem and corollary show that the hardness results obtained for NPbOpt's can indeed be strengthened to quasi-completeness.

Theorem 4.3. Let f be a function such that $f \leq_{1\text{-}T}^{FP} \mathrm{Opt}_{R,c}$, for some NPbOpt (R, c). Then f is in $\mathrm{FP}_{\|}^{\mathrm{NP}}$.

Proof. By Lemma 2.2, we can assume that the reduction is uniform. Let f be reducible to $\mathrm{Opt}_{R,c}$ via t_1, $t_2 \in \mathrm{FP}$, i.e., we have for any x and for all $y \in \mathrm{OptSol}_{R,c}(t_1(x))$ that $f(x) = t_2(x,y)$.

Define NP sets A and B as follows. For any $x \in \Sigma^*$, $k \leq p(|x|)$, and $i \leq q(|x|)$, where p is some polynomial that bounds the solution cost function c and q is some polynomial that bounds the length of the solutions for x with respect to R

$$(x,k) \in A \iff$$

$$\exists y \in S_R(t_1(x)) : c(t_1(x), y) \geq k,$$

and

$$(x,k,i) \in B \iff$$
$$\exists y \in S_R(t_1(x)) : c(t_1(x), y) \geq k \text{ and}$$
$$\text{the } i\text{-th bit of } t_2(x,y) \text{ is a one.}$$

Let k^* be the maximal k such that $(x,k) \in A$. Then the i-th bit of $f(x)$ is one, if $(x,k^*,i) \in B$, and zero, otherwise, for $i = 1,\ldots,q(|x|)$. Therefore, we can compute $f(x)$ by asking in parallel the queries (x,k) to A and (x,k,i) to B, for $k = 1,\ldots,p(|x|)$ and $i = 1,\ldots,q(|x|)$. Thus $f \in \mathrm{FP}_{\|}^{A\oplus B} \subseteq \mathrm{FP}_{\|}^{\mathrm{NP}}$. □

In fact, in Theorem 4.3, it suffices to assume that $f \leq_{tt}^{FP} \mathrm{Opt}_{R,c}$.

Corollary 4.4. Let (R,c) be an universal NPbOpt having an embedding operator. Then we have $\mathrm{FP}_{\|}^{\mathrm{NP}} = \{\, f \mid f \leq_{1\text{-}T}^{FP} \mathrm{Opt}_{R,c} \,\} = \{\, f \mid f \leq_{tt}^{FP} \mathrm{Opt}_{R,c} \,\}$.

It follows that if any $\mathrm{FP}^{\mathrm{NP}}$ complete function is reducible to, say F_{zero}, then this function can already be computed with parallel queries to NP, and hence $\mathrm{FP}^{\mathrm{NP}}$ would be the same as $\mathrm{FP}_{\|}^{\mathrm{NP}}$.

Corollary 4.5. Let (R,c) be an universal NPbOpt having an embedding operator. Then we have $f_{left} \leq_{1\text{-}T}^{FP} \mathrm{Opt}_{R,c} \iff \mathrm{FP}_{\|}^{\mathrm{NP}} = \mathrm{FP}^{\mathrm{NP}} \iff \mathrm{P}^{\mathrm{NP}} = \mathrm{P}_{\|}^{\mathrm{NP}}$.

5 Negative Results and NPSV

For certain subclasses of $\mathrm{FP}_{\|}^{\mathrm{NP}}$, one can show that it is *not* possible to compute satisfying assignments, unless the Polynomial Time Hierarchy PH collapses. Hemaspaandra et al. [HNOS93] showed such a result for the class NPSV.

Definition A nondeterministic Turing transducer N is *single-valued*, if, for each input x, N makes the same output on all accepting computations. *NPSV* is the class of partial functions that can be computed by single-valued NP transducers. $\mathrm{FP}_{\|}^{\mathrm{NPSV}[k]}$ denotes the class of functions that is computable with k nonadaptive queries to an NPSV oracle.

Note that $\mathrm{NPSV} \subseteq \mathrm{FP}_{\|}^{\mathrm{NP}}$, since with the help of an NP set one can get in parallel all the bits of an NPSV function value. In fact, $\mathrm{FP}_{\|}^{\mathrm{NP}} = \mathrm{FP}_{\|}^{\mathrm{NPSV}}$ [FHOS93].

Theorem 5.1. **[HNOS93]** If $\mathrm{NPSV} \cap F_{sat} \neq \emptyset$, then $\mathrm{PH} = \Sigma_2^P$.

Instead of F_{sat}, we note that Theorem 5.1 also holds for F_R, for any universal relation R. The following lemma will enable us to extend this result to $\mathrm{FP}^{\mathrm{NPSV}[1]}$.

Lemma 5.2. Let R be an NP relation. Then we have $F_R \cap \mathrm{FP}^{\mathrm{NPSV}[1]} \neq \emptyset \iff F_R \cap \mathrm{NPSV} \neq \emptyset$.

Proof. If $F_R \cap \mathrm{FP}^{\mathrm{NPSV}[1]} = \emptyset$ then the lemma clearly holds. So assume that $f \in F_R \cap \mathrm{FP}^{\mathrm{NPSV}[1]}$. Let M be a FP machine and N be an NPSV machine witnessing that $f \in \mathrm{FP}^{\mathrm{NPSV}[1]}$. We have to show that $F_R \cap \mathrm{NPSV} \neq \emptyset$.

Consider the following machine N' on input x. First, N' simulates M on input x until M queries it's oracle. Let q_x be the query. Then N' assumes that the answer to the query is $\perp$ and continues the simulation of M. Let y be the output of M. If xRy holds, then N' outputs y and halts. (Note that this is a deterministic computation up to here.) Otherwise, N' guesses an accepting computation for $N(q_x)$. (Note that there must exist an accepting computation now if $x \in D_R$.) Let z be the value computed by $N(q_x)$. Now, N' continues the simulation of M with z as the answer to q_x. Let y' be the output of M. If xRy' holds, then N' outputs y' and halts.

Clearly, N' is an NPSV machine. Furthermore, N' outputs some $y \in S_R(x)$ iff $x \in D_R$. □

Corollary 5.3. Let R be an universal relation. If $F_R \cap \mathrm{FP}^{\mathrm{NPSV}[1]} \neq \emptyset$ then $\mathrm{PH} = \Sigma_2^P$.

It is an interesting open problem to extend Corollary 5.3 to a larger class than $\mathrm{FP}^{\mathrm{NPSV}[1]}$ and still get a collapse of the Polynomial Time Hierarchy.

The following theorem shows that for any universal relation R, F_R is hard, and in fact even quasi-complete for NPSV, $\mathrm{FP}^{\mathrm{NPSV}[1]}$ and $\mathrm{FP}_{\|}^{\mathrm{NPSV}}$ with respect to different types of reductions.

Theorem 5.4. Let R be an universal relation.

(i) $\mathrm{NPSV} = \{\, f \mid f \leq_{1\text{-}T}^{FP\text{-}strict} F_R \,\}$
$= \{\, f \mid f \leq_{tt}^{FP\text{-}strict} F_R \,\}$,

(ii) $\mathrm{FP}^{\mathrm{NPSV}[1]} = \{\, f \mid f \leq_{1\text{-}T}^{FP} F_R \,\}$,

(iii) $\mathrm{FP}_{\|}^{\mathrm{NPSV}} = \{\, f \mid f \leq_{tt}^{FP} F_R \,\}$.

Proof. (i) Let f be in NPSV and let N be an NPSV machine for f. Consider the following NP relation R_N. For $x, y \in \Sigma^*$, where $|y| \leq p(|x|)$ and p is some polynomial that bounds the the running time of N

$$xR_Ny \iff y \text{ is a computation path of } N \text{ on } x \text{ where } N \text{ produces an output.}$$

Since R is universal there exist two FP functions t_1 and t_2 such that t_1 maps any x from the domain of R_N to the domain of R and for any solution y for $t_1(x)$, i.e., $t_1(x)Ry$ holds, $t_2(x,y)$ gives a solution for x, i.e., $xR_Nt_2(x,y)$ holds. Clearly, from $t_2(x,y)$ one can compute $f(x)$ in polynomial time. Furthermore, the reduction (t_1, t_2) is strict.

For the other direction, let f be a function that is $\leq_{tt}^{FP\text{-}strict}$-reducible to F_R via the FP functions t_1 and t_2. Consider the following NP machine N on input x. First, N computes the queries $t_1(x) = (w_1, \ldots, w_k)$ and then guesses solutions $y_1, \ldots, y_k$ for them with respect to R. If w_iRy_i for $i = 1, \ldots, k$, then N outputs $t_2(x, y_1, \ldots, y_k)$.

Since the reduction is strict, there will be a path where N finds solutions for all $w_1, \ldots, w_k$. Furthermore, for every k-tuple of solutions $y_1, \ldots, y_k$, $t_2(x, y_1, \ldots, y_k)$ will give the same value, namely $f(x)$. Hence, N is single-valued and computes f.

The inclusion from left ro right of (ii) and (iii), follow by an easy modification of the argument for (i). In fact, we get more general that $\mathrm{FP}_{\|}^{\mathrm{NPSV}[k]} \subseteq \{ f \mid f \leq_{k\text{-}tt}^{FP} F_R \}$, for every $k \in \mathrm{FP}$.

For the reverse inclusion of (ii), let f be a function that is $\leq_{1\text{-}T}^{FP}$-reducible to F_R via the FP functions t_1 and t_2. Consider the following NP machine N on input x. First, N computes $t_1(x)$ and then guesses a solution y for it with respect to R. If $t_1(x)Ry$, then N outputs $t_2(x,y)$.

Clearly, N is a NPSV machine that outputs $f(x)$ if it is defined. Now a FP machine with N as an oracle can compute $f(x)$ by producing the same output as N on x when it is defined, and $t_2(x, \perp)$, otherwise.

For the reverse inclusion of (iii), let f be a function that is $\leq_{tt}^{FP}$-reducible to F_R via the FP functions t_1 and t_2. We show how to compute f with parallel queries to NP. Let $x \in \Sigma^*$ be fixed and let $w_1, \ldots, w_k$ be the queries produced by $t_1(x)$. By asking the w_i's to D_R, we can find out which ones of them actually have a solution with respect to R. Suppose l of $w_1, \ldots, w_k$ are in D_R, where $0 \leq l \leq k$. Observe that an NP machine knowing l can actually compute (on some path) the w_i's in D_R together with some solution for them, and therefore, via t_2 also $f(x)$. Since there are only k possibilities for l, i.e. polynomially many, we can define an NP set that, for each l, refers to the bits of $f(x)$, similar as in the proof of Theorem 4.3. All together, we can compute $f(x)$ by asking polynomially many queries in parallel to D_R and the latter NP set. □

In other words, for any universal relation R we have shown that F_R is quasi-complete for NPSV and $\mathrm{FP}^{\mathrm{NPSV}[1]}$, but not complete for these classes for $\leq_{1\text{-}T}^{FP\text{-}strict}$- and $\leq_{1\text{-}T}^{FP}$-reduction, respectively, unless the Polynomial Time Hierarchy collapses.

Watanabe and Toda [WT90] showed that if $f_{left} \leq_{tt}^{FP\text{-}strict} F_{sat}$, then NP = co-NP. We strengthen their result by showing the same consequence under a much weaker assumption.

Corollary 5.5. Let R be a universal relation.

(i) If $\mathrm{FP}^{\mathrm{NPSV}[1]} \leq_{tt}^{FP\text{-}strict} F_R$, then NP = co-NP.

(ii) For any $k > 1$, if $\mathrm{FP}_{\|}^{\mathrm{NPSV}[k]} \leq_{1\text{-}T}^{FP} F_R$, then $\mathrm{P}_{\|}^{\mathrm{NP}[l]} = \mathrm{P}^{\mathrm{NP}[1]}$ for any $l \geq 1$, and hence the Polynomial Time Hierarchy collapses.

(iii) Let (R', c) be an universal NPbOpt having an embedding operator. If $\mathrm{Opt}_{R',c} \leq_{1\text{-}T}^{FP} F_R$, then $\mathrm{P}_{\|}^{\mathrm{NP}} = \mathrm{P}^{\mathrm{NP}[1]}$, and hence the Polynomial Time Hierarchy collapses.

Proof. (i) From the assumption together with Theorem 5.4 (i), we conclude that $\mathrm{FP}^{\mathrm{NPSV}[1]} = \mathrm{NPSV}$. As a special case, when considering only characteristic functions, it follows that $\mathrm{P}^{\mathrm{NPSV}[1]} = \mathrm{NP}$. Now, observe that co-NP $\subseteq \mathrm{P}^{\mathrm{NPSV}[1]}$.

(ii) From the assumption together with Theorem 5.4 (ii), we conclude that $\mathrm{FP}_{\|}^{\mathrm{NPSV}[k]} = \mathrm{FP}^{\mathrm{NPSV}[1]}$, and hence $\mathrm{P}^{\mathrm{NPSV}[k]} = \mathrm{P}^{\mathrm{NPSV}[1]}$. Now, the claim follows since $\mathrm{P}_{\|}^{\mathrm{NPSV}[l]} = \mathrm{P}_{\|}^{\mathrm{NP}[l]}$ for any $l \geq 0$ [FHOS93].

(iii) Follows from a similar argument as in (ii) together with Theorem 3.1 and the transitivity of the $\leq_{1\text{-}T}^{FP}$-reduction. □

References

[AB92] M. Agrawal, S. Biswas. Universal Relations. In *Proc. 7th Structure in Complexity Theory Conference*, pages 207–220, 1992.

[Co71] S. Cook. The Complexity of Theorem-Proving Procedures. In *Proc. 3rd ACM Symposium on Theory of Computing*, pages 151–158, 1971.

[CT91] Z. Chen and S. Toda. On the Complexity of Computing Optimal Solutions. In *International Journal of Foundations of Computer Science*, 2, pages 207-220, 1991.

[CT93] Z. Chen and S. Toda. An Exact Characterization of $\mathrm{FP}_{\|}^{\mathrm{NP}}$. Manuscript, 1993.

[FHOS93] S. Fenner, S. Homer, M. Ogiwara, and A. Selman. On Using Oracles That Compute values. In *10-th Annual Symposium on Theoretical Aspects of Computer Science*, pages 398–407, 1993.

[GS88] J. Grollmann and A. Selman. Complexity Measures for Public-Key Cryptosystems. *SIAM Journal on Computing*, 17:309–335, 1988.

[HNOS93] L. Hemaspaandra, A, Naik, M. Ogiwara, and A. Selman. Finding Satisfying Assignments Uniquely Isn't so Easy: Unique Solutions Collapes the Polynomial Hierarchy. Manuscript, 1993.

[HU79] J. Hopcroft and J. Ullman. *Introduction to Automata Theory, Languages, and Computation.* Addison-Wesley, 1979.

[JT93] B. Jenner and J. Torán. Computing Functions With Parallel Queries to NP. In *Proc. 8th Structure in Complexity Theory Conference*, pages 280–291, 1993.

[Ka88] J. Kadin. *Restricted Turing Reducibilities and the Structure of the Polynomial Time Hierarchy.* PhD thesis, Cornell University, February 1988.

[Kr86] M. Krentel. The Complexity of Optimization Problems. In *Proc. 18th ACM Symposium on Theory of Computing*, pages 69–76, May 1986.

[Va76] L. Valiant. The Relative Complexity of Checking and Evaluating. *Information Processing Letters*, 5, pages 20–23, 1976.

[WT90] O. Watanabe and S. Toda. Structural Analysis on the Complexity of Inverse Functions. In *Mathematical Systems Theory*, 26, 203–214, 1993.