Optimization of Unary Costs

Andreas Jakoby

Christian Schindelhauer

Medizinische Universität Lübeck*

March 1997

Abstract

We investigate the computational complexity of optimal solutions, when the costs can be bounded by a polynomial, i.e. can be unary coded. Here, we revisit the computation problem OptP[$\log n$], introduced by [K 88] and the search problem, called $\mathcal{NP}$bOpt by [BKT 94]. In this paper we investigate corresponding classes uMax$\mathcal{P}$ and $\mathcal{NP}$uMax as well as more general formulations of optimization problems: uOpt$\mathcal{P}$ and $\mathcal{NP}$uOpt. These classes turn out to form the metric closure $\Gamma_{1\text{-T}}$ of the maximization classes. For this we introduce adapted reductions, where for instance we tighten the reduction model of search problems to the so-called strong reduction, where the solution space is completely covered by the mapping of the reduction, and show that completeness and closure results still hold.

Many properties of nondeterministic Turing machines (NTM) like time or the number of guesses can be used to describe the class uMax$\mathcal{P}$. For this we refer to Krentel's proof of the correspondence of uMax$\mathcal{P}$ to the output of a metric NTM (here called $\mathcal{NP}$-transducer). Moreover, we show that the class of search problems $\mathcal{NP}$uOpt can be completely characterized by $\mathcal{NP}$-transducers, too.

Furthermore, we show that uOpt$\mathcal{P}$-complete problems are not only complete for $\mathcal{FP}^{\mathcal{NP}[\log]}$ [K 88] but also for $\mathcal{FP}_{tt}^{\mathcal{NP}}$ restricted to functions with unary range, called $\mathcal{UF}$. This gives the new characterization of uOpt$\mathcal{P}$ $=$ $\Gamma_{1\text{-T}}$(OptP[$\log n$]):

$$\mathcal{FP}^{\mathcal{NP}[\log]} \cap \mathcal{UF} \;=\; \text{uOpt}\mathcal{P} \;=\; \mathcal{FP}_{tt}^{\mathcal{NP}} \cap \mathcal{UF}\ .$$

Note that the question, whether $\mathcal{FP}^{\mathcal{NP}[\log]} \neq \mathcal{FP}_{tt}^{\mathcal{NP}}$ is yet still open.

1 Introduction

Most of the classes, investigated in complexity theory, are sets of predicates. Natural problems, however, usually address the difficulty to determine the value of functions, or to give one of many possible solutions that satisfy certain conditions. Moreover, many such problems can be formulated as the problem to optimize some easy computable cost functions.

Since Krentel's study of $\mathcal{NP}$-optimization problems in [K 88] the interest in this topic has steadily been rising. Krentel considered the case that a function underlying the optimization problem is computable in polynomial time. He calls such a function **cost function**, e.g. the cost function of the traveling salesman problem is the total length of a round tour, with respect to the given map. Moreover, he restricts to maximizations problems which include minimization problems, if the cost function is negated.

*Institut für Theoretische Informatik, Wallstraße 40, 23560 Lübeck, Germany
email: jakoby / schindel @ informatik.mu-luebeck.de

There are many optimization problems, where the costs are bounded by the size of the problem itself, e.g. the largest clique of graph is bounded by the graph. Moreover, for these problems one could use a unary encoding of the costs within polynomial time.

Many optimization problems may be encoded in this way, e.g. the chromatic number of a graph, the length of a path, the number of satisfied clauses in a given Boolean formula in conjunctive normal form, time of a nondeterministic Turing machine or worst time of a circuit. Furthermore, any given cost function may be reduced to unary range by an arbitrary polynomial time bounded mapping yielding an approximation of the original cost function. Consider the traveling salesman problem for an example. In general this problem is given with binary weights, but if the problem is given by a map, this gives an inherent unary encoding. On the other hand, the salesman will check the efficiency of the tour in multiples of seconds or kilometers, which the salesman can easily represent unary. We think, unary cost functions often represent the natural case and it will turn out that the situation for unary costs is different than in the general case.

This leads to the class of problems we want to investigate here, i.e. the computation and the search problems of optimizing unary costs. For unbounded costs it is known that the computation problem is as hard as the search problem, which is $\mathcal{FP}^{\mathcal{NP}}$-complete. For the unary optimization problem it is known that there are $\mathcal{FP}^{\mathcal{NP}[\log]}$-complete computation problems [K 88], while corresponding search problems are hard for $\mathcal{FP}_{tt}^{\mathcal{NP}}$.

In this paper we study these classes under the aspect of closure, reducibility and completeness. For this we introduce appropriate reductions. During this study we show that these completeness results lead to the equality of $\mathcal{FP}^{\mathcal{NP}[\log]}$ and $\mathcal{FP}_{tt}^{\mathcal{NP}}$, if both classes are restricted to functions with unary range. Note, that the question whether logarithmic adaptive queries of a $\mathcal{NP}$-oracle equal polynomial non-adaptive $\mathcal{NP}$-oracle queries, is still open for the unrestricted functional classes. For decision problems it holds [W 88]

$$\mathcal{P}_{tt}^{\mathcal{NP}} \;=\; \mathcal{P}^{\mathcal{NP}[\log]}\ .$$

For the functional classes it is known that the collapse $\mathcal{FP}_{tt}^{\mathcal{NP}} = \mathcal{FP}^{\mathcal{NP}[\log]}$ implies $\mathcal{NP} = \mathcal{R}$ and FewP $= \mathcal{P}$ [S 94]. $\mathcal{R}$ is the class of all decision problems solved by polynomial-time random Turing machines, which accepts on more than half of all computations path, or rejects on all paths. FewP is the set of all languages recognizable by polynomial-time NTMs, for which the number of accepting computations is bounded by a polynomial.

Chen and Toda [CT 91] showed that problems like searching the maximum clique, finding a minimum coloring or a maximum path in a graph, are $\mathcal{FP}_{tt}^{\mathcal{NP}}$-hard. On the other hand, for the unary cost search problem no better upper bound than $\mathcal{FP}^{\mathcal{NP}}$ is known yet. Note that these classes form sets of relations, since many inputs may lead to optimal costs.

Buhrmann, Kadin and Thierauf [BKT 94] showed that for the class of search optimization problems $\mathcal{NP}$bOpt those problems are complete with respect to weak metric reduction, which is a form of 1-Turing reduction adapted to sets of relations. Based on the notion of universal relations [AB 92], they present a whole set of problems with this property (universal $\mathcal{NP}$bOpt having an embedding operator).

In their definition they require instance and solution to suffice a $\mathcal{NP}$-relation in advance. We on the contrary consider classes of recursive relations, where one always can embed total functions into the relations. Later on we will see that our definitions and $\mathcal{NP}$bOpt have equal computational complexity.

Furthermore, we want to point out, that many interesting features of non deterministic computation are described by unary cost maximization problems. Later on, we extend the gallery of complete problems for both the search and the computation problems. In all definitions we take special care for the difference of maximization and optimization problems, which include minimization problems. This is necessary, since maximization problems are not closed under metric reductions, unless $\mathcal{NP} =$ co-$\mathcal{NP}$ [VW 95, HW 97].

2 Computing the Optimum of Unary Cost Functions

In this paper we concentrate on unarily valued functions.

Definition 1
The set $\mathcal{UF}$ consists of all functions with unary (tally) range, i.e. $\boldsymbol{\mathcal{UF}} := \{f : \Sigma^* \to 1^*\}$. *Further define* $\boldsymbol{\mathcal{UFP}} := \mathcal{UF} \cap \mathcal{FP}$, *where $\mathcal{FP}$ denotes the set of polynomial time computable functions.*

The computation problem of unary maximization and optimization with respect to these cost functions is defined as follows.

Definition 2
For a given $\mathcal{UFP}$ cost function $c : \Sigma^* \times \Sigma^* \to 1^*$ and a polynomial p a **unary maximization problem** *is to compute the* **maximum cost function** $\boldsymbol{c^*}$*:*

$$\boldsymbol{c^*(x)} := \max\{c(x,y) \mid |y| \leq p(|x|)\} .$$

We call x an **instance**, *y an* **input** *of the problem. If y maximizes $c(x,y)$ we call y a* **witness** *for the optimal solution.*
For a given $\mathcal{UFP}$ cost function $c : \Sigma^* \times \Sigma^* \to 1^*$ and a $\mathcal{FP}$-mapping $m : \Sigma^* \to \Sigma^*$ a **unary optimization problem** *is to compute the* **optimal cost function** $\boldsymbol{m(c^*)}$.
By $\mathbf{uMax\mathcal{P}}$ *we define the set of all unary maximization problems. Further* $\mathbf{uOpt\mathcal{P}}$ *is the set of all unary optimization problems.*

For another characterization of uMax$\mathcal{P}$ we will use the model of $\mathcal{NP}$-transducer (also known as metric Turing machine). In contrast to the usual NTM model the result of a transducer does not depend on accepting or rejecting of a computation, it only depends on the final inscription of the output tape.

Definition 3 *A* $\boldsymbol{\mathcal{NP}}$**-transducer** *is a polynomial time bounded nondeterministic Turing machine that produces an output string on every computation path.*

We allow different outputs to appear on the halting states of a computation. Furthermore, we want to extract a unarily valued function from these outputs. A natural definition for this is the length of the maximum output introduced by [K 88].

Definition 4 *Let* $\mathbf{mol_N(x)}$ *(**m**aximal **o**utput **l**ength) be the maximal length of all outputs of $\mathcal{NP}$-transducer N on input x.*

These machine induced functions give an additional characterization of unary optimization problems.

Theorem 1 *[K 88]* $\quad \text{uMax}\mathcal{P} = \bigcup_{\mathcal{NP}\text{-transducer } N} \text{mol}_N$.

Proof In the notation of Krentel: $\text{OptP}[\log n] = \bigcup_{\text{metric NTM}} \text{opt}^{\text{Univ}[N,\log n]}$. ∎

Surprisingly many unary codable properties of a nondeterministic Turing machine can be characterized with uMax$\mathcal{P}$, e.g. the maximum number of guesses a NTM makes on an accepting path. To express this we state the following definitions of properties of NTMs.

Definition 5 *Let* $\mathbf{comp\text{-}tree_N(x)}$ *denote the computation tree of N. Define* $\mathbf{guess\text{-}tree_N(x)}$ *for a NTM N as the computation tree of a NTM on x reduced to all leafs and internal nodes with fanout at least 2. Let* $\mathbf{guess\text{-}path_N(x)}$ *be the set of all paths of* guess-tree$_N(x)$ *starting at the root and*

ending at a leaf of guess-tree$_N(x)$. *Furthermore define* $\mathbf{acc\text{-}path_N(x)} \subseteq$ guess-paths$_N(x)$ *as the set of all paths of* guess-tree$_N(x)$ *which represent accepting computations of* N *on input* x.

For a path $p \in$ guess-paths$_N(x)$ *let* $\mathbf{|p|}$ *denote the length of* p *and* $\mathbf{p[1], p[2], \ldots, p[|p|]} \in \{0, \ldots, c-1\}^{|p|}$ *be an encoding for the sequence of nondeterministic choices* N *makes (Here* c *denotes the maximum possible number of non-deterministic choices* N *can make in a single step, e.g. derived from the state-transition-graph of* N*).*

In contrast to computation trees, guess trees do not provide any node with fanout one. In fact, by replacing all such nodes (and paths without branches) with edges, computation trees transform into guess trees. Intuitively, a guess tree gives a high level description of computation of a NTM: Every node represents some deterministic computation, whilst an edge gives the nondeterminism of the computation.

W.l.o.g. we restrict our investigation to NTMs N where guess-tree$_M(x)$ is a binary tree with ordered choices. So we get a left (resp. 0) and a right (resp. 1) choice on the computation for the next state. We call a NTM N **admissible**, if guess-tree$_M(x)$ is a complete binary tree.

Definition 6 *For a given nondeterministic Turing machine* N *define the following unary range functions:*

$$\begin{aligned}
\mathbf{time_N(x)} &:= \text{depth}(\text{comp-tree}_N(x)) \ , \\
\mathbf{guess_N(x)} &:= \begin{cases} \max\{|p| \mid p \in \text{acc-paths}_N(x) \} \ , & N(x) \text{ accepts,} \\ 0 \ , & \text{else ,} \end{cases} \\
\mathbf{max\text{-}1\text{-}guess_N(x)} &:= \begin{cases} \max\{\sum_i p[i] \mid p \in \text{acc-paths}_N(x)\} & N(x) \text{ accepts,} \\ 0 \ , & \text{else .} \end{cases}
\end{aligned}$$

Let $\mathbf{max\text{-}reversal_N(x)}$ *be the maximum number of reversals* N *does on an accepting path of input* x *and 0, if* N *rejects on* x.

All these functions are closely related, since they can describe the time behavior of a given NTM.

Lemma 1 *For each polynomial time bounded NTM* N *there exists a polynomial time bounded NTM* N' *such that*

$$\begin{aligned}
\text{guess}_{N'} &= \text{time}_N \ , && (1) \\
\text{max-1-guess}_{N'} &= \text{guess}_N \ , && (2) \\
\text{mol}_{N'} &= \text{max-1-guess}_N \ , && (3) \\
\text{max-reversal}_{N'} &= \text{time}_N \ , && (4) \\
\text{mol}_{N'} &= \text{max-reversal}_N \ , && (5) \\
\text{time}_{N'}(x) &= (2 \cdot \text{mol}_N(x) + 1) 2^{\log p_N(|x|)} \ , \text{ for a polynomial } p_N \text{ and all } x. && (6)
\end{aligned}$$

(2) also holds if we restrict N' *to admissible NTMs.*

Proof: (1) Define a NTM N' as follows. In the first phase N' simulates N step by step and for each computation path N' counts the number of steps s and the number of choices c. In the second phase N' makes $s - c$ nondeterministic guesses and accepts.

(2) Based on N we construct N' within two steps. First we construct a NTM N'' by replacing every nondeterministic state of N by three nondeterministic states of N'', such that there exists a path $p \in$ acc-paths$_N(x)$ iff there exists a path $q \in$ acc-paths$_{N''}(x)$ such that

$$\forall i \in \{1, \ldots, |p|\}: \quad p[i] = 0 \Leftrightarrow q[2i-1]q[2i] = 01 \ \text{ and } \ p[i] = 1 \Leftrightarrow q[2i-1]q[2i] = 10 \ .$$

Note that $\text{guess}_N(x) = \max_{q \in \text{acc-paths}_{N''}(x)} |\{i < |q| \mid q[i] = 1\}| = \text{guess}_{N'}(x)/2$ if acc-paths$_N(x) \neq \emptyset$.

Now N' is an admissible NTM that simulates N and accepts on a path $p \in \text{acc-paths}_{N'}(x)$ iff there exists a path $q \in \text{acc-paths}_{N''}(x)$ such that

$$\forall i \in \{1, \ldots, |q|\}: \quad p[i] = q[i] \qquad \text{and} \qquad \forall i \in \{|q|+1, \ldots, |p|\}: \quad p[i] = 0 \; .$$

Hence, all paths of $\text{guess-paths}_{N'}(x)$ have same length and if $\text{acc-paths}_N(x) \neq \emptyset$ it holds

$$\text{guess}_N(x) = \max_{q \in \text{acc-paths}_{N'}(x)} |\{i < |q| \mid q[i] = 1\}| \; .$$

(3) N' simulates N step by step and counts on each computation path p the number of right choices c of N on p. If N accepts the input on computation path p, then N' outputs 1^c. Otherwise N' outputs the empty string λ.

(4) Define a NTM N' as follows. In the first phase N' simulates N step by step and counts the number of steps s and reversals r. In the second phase N' makes $s - r$ reversals and accepts.

(5) N' simulates N step by step and counts on each computation path p the number of reversals r of N on p and 1^r.

(6) Let p_N be a polynomial that bounds the time of N with respect to the input length n. First, N' simulates N step by step and counts the number of steps t for each computation path. If N outputs a string of length s, then N' makes $(2 \cdot s + 1)2^{\log p_N(|x|)} - t$ additional steps before accepting. ∎

Reductions between maximization problems require monotone transformations of the cost functions. The according definition of such functions in $\mathcal{UFP}$ will be called $\mathcal{UMFP}$.

Definition 7 *$\mathcal{UMFP}$ denotes the set of all monotone increasing functions of $\mathcal{UFP} := \mathcal{UF} \cap \mathcal{FP}$.*

Now it follows that mol and therefore $\text{uMax}\mathcal{P}$ are closed under monotone transformation.

Lemma 2

$$\text{uMax}\mathcal{P} \;=\; \bigcup_{f \in \mathcal{UMFP}} \; \bigcup_{\mathcal{NP}\text{-transducer } N} f \circ \text{mol}_N \; .$$

Proof: $\supseteq$: Let $\mathcal{NP}$-transducer N' simulate $\mathcal{NP}$-transducer N on every branch. Then N' computes $f(1^s)$ on every leaf of the computation tree, where $f \in \mathcal{UMFP}$ is given and 1^s the output of N'. ∎

For the class $\text{uOpt}\mathcal{P}$ we do not need such monotone restrictions, since $\text{uOpt}\mathcal{P}$ is closed even under the application of arbitrary $\mathcal{UFP}$-functions.

Theorem 2 *For any $g_N \in \{\text{guess}_N, \text{time}_N, \text{max-1-guess}_N, \text{mol}_N, \text{max-reversal}_N\}$ holds*

$$\bigcup_{f \in \mathcal{UMFP}} \; \bigcup_{N \in \text{NTM}} f \circ g_N \;\subseteq\; \text{uMax}\mathcal{P} \quad \textit{and} \quad \text{uOpt}\mathcal{P} \;=\; \bigcup_{f \in \mathcal{UFP}} \; \bigcup_{N \in \text{NTM}} f \circ g_N \; .$$

Proof: $\text{uOpt}\mathcal{P} \subseteq \bigcup_{f \in \mathcal{UFP}} \bigcup_{N \in \text{NTM}} f \circ g_N$ follows from the fact, that all transformation given in lemma 1 are invertible. ∎

3 Complete Problems for $\text{uOpt}\mathcal{P}$

In the following, we investigate the structure of $\text{uMax}\mathcal{P}$ and $\text{uOpt}\mathcal{P}$ with respect to completeness and closure. The appropriate form of reduction for these classes as well as for many other functional classes, such as $\mathcal{FP}_{tt}^{\mathcal{NP}}$ or $\mathcal{FP}^{\mathcal{NP}}$, is the metric reduction.

Definition 8 *For functions f, g the* ***1-Turing reduction*** *(also known as metric reduction) $f \leq_{1\text{-}T}^{\mathcal{FP}} g$ is defined as*

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

$\Gamma_{1\text{-}T}(\mathcal{C})$ denotes the closure of a set of functions $\mathcal{C}$ under 1-Turing reductions.

For formal reasons $\text{uOpt}\mathcal{P}$ cannot be closed under metric reductions, since its range may not be unarily coded anymore. Therefore we restrict the metric reduction onto $\mathcal{UF}$.

Definition 9 *For functions f, g a* ***unary range 1-Turing reduction*** *is $f \leq_{1\text{-}T}^{\text{u}\mathcal{FP}} g$ is a metric reduction with $t_2 \in \mathcal{UF}$.*
If t_2 is monotone w.r.t. its second parameter, i.e. $\forall x, y_1, y_2 \; y_1 < y_2 \implies t_2(x, y_1) \leq t_2(x, y_2)$, we define $f \leq_{1\text{-}T}^{\text{um}\mathcal{FP}} g$ as ***unary range monotone 1-Turing reduction.***

In contrast to $\leq_{\text{1-T}}^{\text{um}\mathcal{FP}}$ the $\leq_{\text{1-T}}^{\text{u}\mathcal{FP}}$-reduction does not restrict the metric reduction for unarily valued functions, since for $f, g \in \mathcal{UF}$ it holds $f \leq_{\text{1-T}}^{\text{u}\mathcal{FP}} g \Leftrightarrow f \leq_{\text{1-T}}^{\mathcal{FP}} g$.

Theorem 3

$$\begin{aligned} \Gamma_{1\text{-}T}^{\text{u}\mathcal{FP}}(\text{uMax}\mathcal{P}) &= \Gamma_{1\text{-}T}^{\text{u}\mathcal{FP}}(\text{uOpt}\mathcal{P}) = \text{uOpt}\mathcal{P} \; , \\ \Gamma_{1\text{-}T}^{\text{um}\mathcal{FP}}(\text{uMax}\mathcal{P}) &= \text{uMax}\mathcal{P} \; . \end{aligned}$$

Proof: $\Gamma_{\text{1-T}}^{\text{u}\mathcal{FP}}(\text{uOpt}\mathcal{P}) \subseteq \text{uOpt}\mathcal{P}$:
Given an optimization problem $m \circ c^* \in \text{uOpt}\mathcal{P}$ and let $f \leq_{\text{1-T}}^{\text{u}\mathcal{FP}} m \circ c^*$, where t_1, t_2 denote the functions of the reductions. We prove that $f \in \text{uOpt}\mathcal{P}$ by constructing a cost function c_f and a mapping m_f, such that $f = m_f \circ c_f^*$.

Since $c, m, t_1, t_2 \in \mathcal{FP}$, there exists a polynomial p such that $t_2(x, m(c^*(t_1(x)))) + c^*(t_1(x)) < p(|x|)$. Let

$$c_f(x, y) \; := \; p(|x|)^2 + p(|x|) \cdot c(t_1(x), y) + t_2(x, m(c(t_1(x), y))) \; .$$

So c_f is maximal for y, iff c is maximal for y. Now, define $m_f(n) \; := \; n \bmod \lfloor \sqrt{n} \rfloor$. Since this functions outputs $t_2(x, m(c(t_1(x), y))$ for $n = c_f(x, y)$ the claim follows.

$\text{uOpt}\mathcal{P} \subseteq \Gamma_{\text{1-T}}^{\text{u}\mathcal{FP}}(\text{uMax}\mathcal{P})$ follows by theorem 2.

$\Gamma_{\text{1-T}}^{\text{um}\mathcal{FP}}(\text{uMax}\mathcal{P}) \subseteq \text{uMax}\mathcal{P}$:
Given $c^* \in \text{uMax}\mathcal{P}$, let $f \leq_{\text{1-T}}^{\text{um}\mathcal{FP}} c^*$, where t_1, t_2 denote the functions of the reductions. $f \in \text{uMax}\mathcal{P}$, since the cost function is directly given by $c_f(x, y) \; = \; t_2(x, c(t_1(x), y)))$. ∎

Note that $\text{uMax}\mathcal{P}$ is not closed under metric reductions unless $\mathcal{NP} = \text{co-}\mathcal{NP}$, since $\text{uMax}\mathcal{P}$ restricted to predicates equals $\mathcal{NP}$ and $\text{uOpt}\mathcal{P}$ contains co-$\mathcal{NP}$. It turns out that many unary computation problems are complete for $\text{uMax}\mathcal{P}$ or $\text{uOpt}\mathcal{P}$. We will give a selection here.

Definition 10 *Define the following optimization problems.*

- `MAX-SAT`*: For a given CNF F, compute the maximum number of satisfiable clauses.*
- `MAX-`k`-SAT`*: For a given k-CNF F, compute the maximum number of satisfiable clauses.*
- `COLORS`*: For a given graph G, compute the chromatic number of G.*
- `MAX-CLIQUE-SIZE`*: For a given graph G, compute the maximum clique size in G.*
- `MAX-PATH-LENGTH`*: Given graph G and two nodes u, v, compute the length of the maximum path from u to v.*
- `MAX-CIRCLE-SIZE`*: Given graph G, compute the size of the longest closed path.*

- CWC-TIME*: Given a circuit C, compute worst time $t_{\mathtt{wc}}$ this circuit can achieve.*
- UW-TSP-LENGTH*: Given a graph G with unary edge weights, compute the length of the shortest round tour.*

For all these problems there exist straight-forward constructions of unary cost functions. Therefore the upper bound uMax$\mathcal{P}$, resp. uOpt$\mathcal{P}$ follows immediately. In [K 88] some of these problems were shown to be uMax$\mathcal{P}$-complete. On the other side, Chen and Toda [CT 91] showed $\mathcal{FP}_{tt}^{\mathcal{NP}}$-hardness of some of the corresponding search problems. Unfortunately the reduction for search problems do not work for the computation problems in general, since we have to present an explicit polynomial time bounded mapping between the optimal costs of the problems.

Lemma 3

$$\begin{array}{rcl|rcl}
\texttt{MAX-1-GUESS} & \leq^{\mathrm{um}\mathcal{FP}}_{1\text{-}T} & \texttt{CWC-TIME} & & & \\
\texttt{MAX-SAT} & \leq^{\mathrm{um}\mathcal{FP}}_{1\text{-}T} & \texttt{MAX-3-SAT} & \leq^{\mathrm{um}\mathcal{FP}}_{1\text{-}T} & \texttt{MAX-CIRCLE-SIZE} \\
 & \leq^{\mathrm{um}\mathcal{FP}}_{1\text{-}T} & \texttt{MAX-PATH-LENGTH} & \leq^{\mathrm{u}\mathcal{FP}}_{1\text{-}T} & \texttt{UW-TSP-LENGTH} \\
\texttt{MAX-3-SAT} & \leq^{\mathrm{um}\mathcal{FP}}_{1\text{-}T} & \texttt{MAX-2-SAT} & & \\
\texttt{MAX-3-SAT} & \leq^{\mathrm{um}\mathcal{FP}}_{1\text{-}T} & \texttt{MAX-CLIQUE-SIZE} & & \\
\texttt{MAX-SAT} & \leq^{\mathrm{u}\mathcal{FP}}_{1\text{-}T} & \texttt{COLORS} & &
\end{array}$$

Proof:

- MAX-1-GUESS $\leq^{\mathcal{FP}}_{1\text{-}\mathrm{T}}$ CWC-TIME
  W.l.o.g. we restrict nondeterministic Turing machines of the MAX-1-GUESS problem to admissible NTMs. Now the reduction follows the construction of the circuit for a given nondeterministic polynomial time bounded Turing machine in theorem 12 of [JS 96].
- MAX-SAT $\leq^{\mathrm{um}\mathcal{FP}}_{1\text{-}\mathrm{T}}$ MAX-3-SAT (Sketch)
  This proof is an adaption of a known reduction from SAT to 3-SAT. Every clause of SAT is mapped to a sub-formula where special care is taken for the maximum number of satisfying clauses in each sub-formula.
- MAX-3-SAT $\leq^{\mathrm{um}\mathcal{FP}}_{1\text{-}\mathrm{T}}$ MAX-CIRCLE-SIZE see [K 88]
- MAX-CIRCLE-SIZE $\leq^{\mathrm{um}\mathcal{FP}}_{1\text{-}\mathrm{T}}$ MAX-PATH-LENGTH (Sketch)
  Let n be the number of nodes in the graph G for the MAX-CIRCLE-SIZE problem. Now we make n disjunct copies G_i of G and add two chains. For every subgraph G_i the node v_i is connected to the end node of a chain of length n. Every neighbor of v_i is connected to the beginning of the other chain of length n. The longest path has to include both chains and a path in one sub-circuit that indicates the longest circle of G.
- MAX-PATH-LENGTH$\leq^{\mathrm{u}\mathcal{FP}}_{1\text{-}\mathrm{T}}$UW-TSP-LENGTH (Sketch)
  The unarily weighted traveling salesman graph consists of three parts: First a copy of the original graph G of the MAX-PATH-LENGTH-problem with n nodes, a circle of length of n and two separate nodes. The graph is complete and the structure given by the unary weight of the edges. The weight can be chosen such that the optimal traveling sales tour in G is equivalent to the longest path.
- MAX-3-SAT $\leq^{\mathrm{um}\mathcal{FP}}_{1\text{-}\mathrm{T}}$ MAX-2-SAT (Sketch)
  Every clause C of the given 3-CNF F can be reduced to a set S of 21 2-CNF clauses, such that if an assignment X of F satisfies C, exactly 18 clauses of S are satisfiable. Otherwise, if C is not satisfied, all assignments corresponding to X can only satisfy 15 clauses of S.
- MAX-3-SAT $\leq^{\mathcal{FP}}_{1\text{-}\mathrm{T}}$ MAX-CLIQUE-SIZE
  The reduction follows a well known $\mathcal{NP}$-reduction of the CLIQUE problem ([AHU 74, K 88]).

- MAX-SAT $\leq^{\mathrm{u}\mathcal{FP}}_{1\text{-}\mathrm{T}}$ COLORS: see [K 88] ∎

Theorem 4 *All problems of definition 9 are* uOpt$\mathcal{P}$ *-complete under* $\leq^{\mathrm{u}\mathcal{FP}}_{1\text{-}T}$*-reducibility. All maximization problems of this definition are* uMax$\mathcal{P}$ *-complete under* $\leq^{\mathrm{um}\mathcal{FP}}_{1\text{-}T}$*-reducibility.*

The following theorem shows that uOpt$\mathcal{P}$ complete problems are also hard for a restricted class of $\mathcal{FP}^{\mathcal{NP}}_{tt}$.

Theorem 5 MAX-1-GUESS *is* $\left(\mathcal{FP}^{\mathcal{NP}}_{tt} \cap \mathcal{UF}\right)$*-complete under* $\leq^{\mathrm{u}\mathcal{FP}}_{1\text{-}T}$*-reducibility.*

Proof: is given in the appendix. ∎

On the other hand, max-1-guess$_N(x)$ can also be determined in polynomial time by binary search using log(size(C)) adaptive queries to an $\mathcal{NP}$-oracle that decides for a given NTM N, an input x, and a number t whether max-1-guess$_N(x) \geq t$.

Corollary 1 $$\mathcal{FP}^{\mathcal{NP}[\log]} \cap \mathcal{UF} \quad = \quad \mathrm{uOpt}\mathcal{P} \quad = \quad \mathcal{FP}^{\mathcal{NP}}_{tt} \cap \mathcal{UF}\ .$$

So it is clear that uOpt$\mathcal{P}$ restricted to predicates equals $\mathcal{P}^{\mathcal{NP}[\log]}$. Note that uMax$\mathcal{P}$ and uMin$\mathcal{P}$, defined according to uMax$\mathcal{P}$, restricted to predicates give the classes $\mathcal{NP}$, resp. co-$\mathcal{NP}$ [VW 95]. Hence, it is not sufficient to consider optimization classes as the union of minimization and maximization problems, since these do not cover the whole range of optimization.

Note that these results do not imply that the uOpt$\mathcal{P}$-problems provide hard functions for $\mathcal{FP}^{\mathcal{NP}}_{tt}$ or $\mathcal{FP}^{\mathcal{NP}[\log]}$. However, Krentel has shown that MAX-SAT is complete for uMax$\mathcal{P}$ and $\mathcal{FP}^{\mathcal{NP}[\log]}$ under metric reducibility.

Theorem 6 *[K 88]* MAX-SAT *is* $\mathcal{FP}^{\mathcal{NP}[\log]}$ *-complete under* $\leq^{\mathcal{FP}}_{1\text{-}T}$*-reducibility.*

Since $\mathcal{FP}^{\mathcal{NP}[\log]}$ is closed under metric reductions, the following holds.

Corollary 2 $$\Gamma_{1\text{-}T}(\mathrm{uOpt}\mathcal{P}) \quad = \quad \mathcal{FP}^{\mathcal{NP}[\log]}\ .$$

This corollary shows that uOpt$\mathcal{P}$ is closely related to $\mathcal{FP}^{\mathcal{NP}[\log]}$, i.e. essentially uOpt$\mathcal{P}$ and $\mathcal{FP}^{\mathcal{NP}[\log]}$ address the same computational complexity. Reducing $\mathcal{FP}^{\mathcal{NP}[\log]}$ to unarily valued functions does not decrease its computational power. On the other hand, $\mathcal{FP}^{\mathcal{NP}[\log]} \neq \mathcal{FP}^{\mathcal{NP}}_{tt}$ implies that $\mathcal{FP}^{\mathcal{NP}}_{tt} \not\leq^{\mathcal{FP}}_{1\text{-}\mathrm{T}} \mathcal{FP}^{\mathcal{NP}}_{tt} \cap \mathcal{UF}$.

4 $\mathcal{NP}$-Optimization Search Problems

Considering search problems we will not distinguish between different solutions having same costs. Note that an additional weighting, like the lexicographically ordering of solutions, may lead to $\mathcal{FP}^{\mathcal{NP}}$-hardness. Furthermore, restricting optimization to a unary cost function leads to complexity classes consisting of relations instead of functions.

In this paper, we consider only recursive relations which implies that all relations are total:

Definition 11 *A relation* $R : \Sigma^* \times \Sigma^*$ *is* ***total*** *iff for all* $x \in \Sigma^*$ *there exists* $y \in \Sigma^*$ *with* $(x, y) \in R$. *A relation* $R' : \Sigma^* \times \Sigma^*$ *solves the relation* $R : \Sigma^* \times \Sigma^*$ *iff* $R' \subseteq R$ *and* R' *is total. If* R' *denotes a function* f*, we call* f *a* ***solution function*** *of* R*, i.e.* $\{(x, f(x)) \mid x \in \Sigma^*\} \subseteq R$.

Based on unary cost functions we will now introduce two classes of search problems, which are closely related to the classes *NPOP* [CT 91] and $\mathcal{NP}$bOpt [BKT 94].

Definition 12 *The* ***unary search maximization problem*** *for a cost function $c \in \mathcal{FP}$ and a polynomial p is the relation*

$$\mathrm{Max}_{c,p} := \{(x,y) \mid c(x,y) = c^*(x) \wedge |y| \leq p(|x|)\} .$$

$\boldsymbol{\mathcal{NP}}$**uMax** *is the class of search maximization problems.*

Note that for a unary search maximization problem a pair (x,y) completely witnesses the optimum of the unary cost function. Investigating a more general form of search problems, we have to consider that a solution y may not completely witness the optimum. Some partial information may not be interesting, e.g. consider a traveling salesman problem, where one only wants to know the next station of an optimal tour.

Definition 13 *Let $\langle y,z\rangle$ denote a polynomial time computable bijective length preserving encoding of y and z.*
The ***unary search optimization problem*** *for a cost function $c \in \mathcal{FP}$ and a polynomial p is the relation*

$$\mathrm{Opt}_{c,p} := \{(x,y) \mid \exists z\, c(x,\langle y,z\rangle) = c^*(x) \wedge |y|+|z| \leq p(|x|)\} .$$

$\boldsymbol{\mathcal{NP}}$**uOpt** *denotes the class of search optimization problems with cost function $c \in \mathcal{UF}$.*

Similar to the unary maximization problem we will now present an exact characterization of $\mathcal{NP}$uOpt based on the output of a transducer.

Definition 14 *For a polynomial time bounded transducer N we define the relation* $\mathbf{mos_N}$ *as the set of all pairs (x,y), such that there exists a computation path of N on input x, where the output is either $y01^\ell$ with $|y|+\ell+1 = \mathrm{mol}_N(x)$ or $1^{\mathrm{mol}_N(x)}$ which implies $y = \lambda$.*

So every computation path providing an output of maximal length indicates an element of the relation mos_N.

Theorem 7 $\quad \mathcal{NP}\mathrm{uOpt} = \bigcup_N \mathrm{mos}_N .$

Proof: $\subseteq$: A $\mathcal{NP}$-transducer N can compute a $\mathcal{NP}$uOpt problem $\mathrm{Opt}_{c,p}$ as follows: On input x N guesses $\langle y,z\rangle$ and outputs $y01^{c(x,\langle y,z\rangle)+p(|x|)-|y|}$.

$\supseteq$: Let z encode a guess path of the $\mathcal{NP}$-transducer N. Now define a polynomial p such that $p(|x|) \geq 2\cdot\mathrm{time}_N(x)$. Let $g(N,x,z)$ denote the output of N on input x and guess path z and define the cost function c as follows

$$c(x,\langle y,z\rangle) := \begin{cases} 1+|g(N,x,z)| , & \text{if } g(N,x,z) \in y01^* , \\ 1+|g(N,x,z)| , & \text{if } y = \lambda \wedge g(N,x,z) \in 1^* , \\ 0 , & \text{else.} \end{cases}$$

∎

Based on an $\mathcal{NP}$-relation R with domain $D_R = \{x \mid \exists y \in \Sigma^* : (x,y) \in R\}$ and a *unary optimal solution cost function* c^* Buhrman et al. introduce the class of *polynomially bounded $\mathcal{NP}$ optimization problems* $\boldsymbol{\mathcal{NP}}$**bOpt** [BKT 94]. They call a partial function f a solution function of $\mathcal{NP}$bOpt, if

$$\forall x \in \Sigma^* : f(x) = \begin{cases} \text{some } y \text{ such that } (x,y) \in \mathrm{Max}_{c,p} , & \text{if } x \in D_R , \\ \perp , & \text{elsewhere .} \end{cases}$$

Note that in our setting D_R is integrated into the cost function such that for example $\perp$ is represented by 0 and all other costs c by $c+1$. Since a solution of a $\mathcal{NP}$bOpt problem depends on the domain of a relation R, a reduction between two $\mathcal{NP}$bOpt problems has also to consider the domains of their relations. [BKT 94] use the notion of weak reduction for $\mathcal{NP}$bOpt, which we adapt for recursive relations as follows:

Definition 15 *A relation F can be reduced to a relation G ($F \leq_{1\text{-}T}^{\text{weak-}\mathcal{FP}} G$) by an 1-Turing reduction, iff for all solution functions $g \subseteq G$ there exists a solution function $f \subseteq F$ such that $f \leq_{1\text{-}T}^{\mathcal{FP}} g$, i.e. for all solution functions $g \subseteq G$*

$$\exists\, t_1, t_2 \in \mathcal{FP}\ \forall\, x \in \Sigma^* \quad : \quad (x, t_2(x, g(t_1(x)))) \in F$$

Considering the weak reduction $\mathcal{NP}$bOpt and $\mathcal{NP}$uMax are equivalent and define the the same complexity class. Note that the weak reduction does not guarantee that all solutions of the reduced relation can occur. To consider such settings we introduce a stricter reduction, the *strong 1-Turing reduction*, which is incomparable to the notion of strict reduction [BKT 94], which refers to properties concerning D_R.

Definition 16 *The strong 1-Turing reduction between relation F and G ($F \leq_{1\text{-}T}^{\text{strong-}\mathcal{FP}} G$) will be defined as follows:*

$$F \leq_{1\text{-}T}^{\text{strong-}\mathcal{FP}} G :\Longleftrightarrow \exists t_1, t_2 \in \mathcal{FP} \quad \forall x \quad t_2(x, \{y \mid (t_1(x), y) \in G\}) \quad = \quad \{z \mid (x, z) \in F\}\ .$$

So, if a relation G and a concrete strong reduction $F \leq_{1\text{-T}}^{\text{strong-}\mathcal{FP}} G$ is given, the relation F is well-defined. Note that a weak reduction may only give a solution $F' \subset F$.

Similar to the problems investigated in section 2, define:

Definition 17 *According to the unary cost optimization problems of definition 6 and 9, we define the search problems:* `MAX-GUESS-PATH`, `MAX-SAT-ASS`, `MAX-2-SAT-ASS`, `MAX-3-SAT-ASS`, `COLORATION`, `MAX-CLIQUE`, `MAX-PATH`, `MAX-CIRCLE`, `UW-TSP`. *Furthermore, we define*
`CWC`*: Given a circuit C, compute worst case input for this circuit,*
`TIME-PATH`*: Given a NTM M and input x, compute a computation path with length* $\text{time}_M(x)$.

Theorem 8
The search problems of Definition 17 are $\mathcal{NP}$uOpt-complete under $\leq_{1\text{-}T}^{\text{strong-}\mathcal{FP}}$-reducibility.

Proof: follows by careful adaption of the reductions concerning the unary cost functions given above. ∎

It is already proved (see for example [CT 91] and [BKT 94]) that most of these search problems are $\mathcal{FP}_{tt}^{\mathcal{NP}}$-hard or uMax$\mathcal{P}$-complete under $\leq_{1\text{-T}}^{\text{weak-}\mathcal{FP}}$-reducibility. On the other hand, no better bound than $\mathcal{FP}^{\mathcal{NP}}$ is known for $\mathcal{NP}$uOpt. Hence,

$$\mathcal{FP}_{tt}^{\mathcal{NP}} \ \subseteq\ \mathcal{NP}\text{uMax} \ \subseteq\ \mathcal{NP}\text{uOpt} \ \subseteq\ \mathcal{FP}^{\mathcal{NP}}\ .$$

Although the optimization problems may conceal some information needed to witness a solution, we can strongly reduce $\mathcal{NP}$uOpt to $\mathcal{NP}$uMax.

Theorem 9 $\quad \mathcal{NP}\text{uOpt} \quad \leq_{1\text{-}T}^{\text{strong-}\mathcal{FP}} \quad \mathcal{NP}\text{uMax}\ .$

Proof: We will show that for all $F \in \mathcal{NP}$uOpt we can construct $G \in \mathcal{NP}$uMax such that $F \leq_{1\text{-T}}^{\text{strong-}\mathcal{FP}} G$. Let $\text{Opt}_{c,p} = F$ and define the cost function c_G of G as

$$c_G(x, \langle y_1, y_2\rangle) \ :=\ c(x, \langle y_1, y_2\rangle)\ .$$

The reduction is given by the functions $t_1(x) := x$ and $t_2(x, \langle y_1, y_2\rangle) := y_1$. The reduction is strong, since for all $x \quad t_2(x, \{\langle y_1, y_2\rangle \mid (x, \langle y_1, y_2\rangle) \in \text{Max}_{c_G,p}\} \ =\ \{y_1 \mid \exists y_2\ c(x, \langle y_1, y_2\rangle) \ =\ c^*(x)\}\ .$ ∎

In addition to $\mathcal{NP}$bOpt, the classes $\mathcal{NP}$uMax and $\mathcal{NP}$uOpt can be strongly reduced to each other. Furthermore, we show that $\mathcal{NP}$uOpt is closed under this reducibility.

Theorem 10 $\Gamma^{\text{strong-}\mathcal{FP}}_{1\text{-}T}(\mathcal{NP}\text{uMax}) = \Gamma^{\text{strong-}\mathcal{FP}}_{1\text{-}T}(\mathcal{NP}\text{uOpt}) = \mathcal{NP}\text{uOpt}$.

Proof of $\Gamma^{\text{strong-}\mathcal{FP}}_{1\text{-T}}(\mathcal{NP}\text{uMax}) = \mathcal{NP}\text{uOpt}$: Let $\text{Max}_{c,p} \in \mathcal{NP}\text{uMax}$, $F \leq^{\text{strong-}\mathcal{FP}}_{1\text{-T}} \text{Max}_{c,p}$, and let t_1, t_2 denote the functions of this reduction. Define the unarily valued cost function

$$c_F(x, \langle y, z\rangle) := \begin{cases} 1 + c(t_1(x), y)\ , & \text{if } y = t_2(z)\ , \\ 0\ , & \text{else}\ . \end{cases}$$

Let $p_F(n) \geq |t_2(n)| + n$, then $F = \text{Opt}_{c_F, p_F} \in \mathcal{NP}\text{uOpt}$. ∎

5 Conclusion

Krentel et al. [K 88, GKR 95] characterized optimization problems by the number $f(n)$ of adaptive $\mathcal{NP}$-oracle queries $\text{OptP}[f(n)] \subseteq \mathcal{FP}^{\mathcal{NP}[f(n)]}$. Moreover, they addressed complete problems in $\text{OptP}[f(n)]$ for $\mathcal{FP}^{\mathcal{NP}}[f(n)]$, where the number of oracles gives a bound for the range of the cost functions as well. These classes are of crucial interest for proving non-approximability of optimization problems, since

Theorem 11 *[K 88] If* Π *is* $\text{OptP}[f(n)]$*-complete and suppose* $\mathcal{P} \neq \mathcal{NP}$ *. Then there exists* ϵ*, such that any polynomial time approximation algorithm* A *must have infinitely many inputs* I *with*

$$|A(I) - \Pi(I)| \geq \tfrac{1}{2} \cdot 2^{f(n^\epsilon)}\ .$$

Here we have considered the case $f(n) = \log n$ and have shown that in addition this class even contains complete problems for $\mathcal{FP}^{\mathcal{NP}}_{tt} \cap \mathcal{UF}$, which gives

$$\Gamma_{1\text{-T}}(\text{OptP[log]}) = \text{uOpt}\mathcal{P} = \mathcal{FP}^{\mathcal{NP}}_{tt} \cap \mathcal{UF}\ .$$

Note that $\text{uMax}\mathcal{P} = \text{OptP[log]}$ is not closed under metric reductions, unless $\mathcal{NP} \neq \text{co-}\mathcal{NP}$. So closure for $\text{uMax}\mathcal{P}$ can only be stated if metric reductions are considered, that use a monotone mapping for the costs. With respect to this reduction, most of the known $\text{uMax}\mathcal{P}$-complete problems stay complete, since their reductions obey this restriction anyway.

For the search problems we mostly referred to Buhrman et al. [BKT 94]. They have defined the relation complexity class $\mathcal{NP}\text{bOpt}$, which essentially is equivalent to the class $\mathcal{NP}\text{uMax}$ defined here. We examined the more general case, if one does not search a total witness of an optimum, but is satisfied with a partial answer. We show that this class $\mathcal{NP}\text{uOpt}$ forms the closure of $\mathcal{NP}\text{uMax}$. For this we use a stronger version of metric reductions as used in [BKT 94]. This weak reduction enables for a given solution function only to compute a solution function for the reduced relation. The strong reduction claims that this solution covers all solutions of the relation.

For a comparison of the computational complexity between search and computation problems of unary cost functions, a straight-forward observation leads to

$$\mathcal{NP}\text{uOpt} =^{\mathcal{FP}}_{1\text{-T}} \text{uOpt}\mathcal{P} \implies \mathcal{FP}^{\mathcal{NP}}_{tt} = \mathcal{FP}^{\mathcal{NP}[\log]} \implies \mathcal{NP} = \mathcal{R} \wedge \text{FewP} = \mathcal{P}\ .$$

This follows from the $\mathcal{FP}^{\mathcal{NP}}_{tt}$-hardness of $\mathcal{NP}\text{uOpt}$-problems [CT 91], the inclusion $\text{uMax}\mathcal{P} \subseteq \mathcal{FP}^{\mathcal{NP}[\log]}$ [K 88, GKR 95], and results from [S 94].

6 Open Problems

Buhrman et al. formulated a general criterion for the completeness of $\mathcal{NP}\text{bOpt}$-functions: universality of the underlying $\mathcal{NP}$-decision problem and the existence of an embedding operator for the cost

function. This criterion does not imply hardness for the computation problem. Assuming that some special sort of one-way functions exist, we are able to prove that such a criterion does not exist, if it does not address the exact values of the cost functions. It is open whether regular one-way-function imply this, too.

The overall open problem of optimization problems is a lower bound criterion for tight non-approximability results. Results using probabilistic checkable proofs [FGLSS 96, H 96] sound promising. Perhaps, these ideas in combination with some techniques shown here, may lead to a general theory of approximability of optimization problems.

Acknowledgment

We want to thank Stephen Fenner, Rüdiger Reischuk, Gerhard Buntrock, Karin Genther, Barbara Goedecke, and Stephan Weis for fruitful discussions and hints to literature.

References

[AB 92] M. Agrawal, S. Biswas, *Universal Relations*, Proc. Structure in Complexity Theory, 1992, pp. 207-220.

[AHU 74] A. Aho, J. Hopcroft, And J. Ullman, *The Design and Analysis of Computer Algorithms*, Addison-Wesley, Reading, MA, 1974.

[BKT 94] H. Buhrman, J. Kadin, Th. Thierauf, *On Functions Computable with Nonadaptive Queries to NP*, Proc. Structure in Complexity Theory, 1994, pp. 43-52.

[CT 91] Z. Chen, S. Toda, *On the Complexity of Computing Optimal Solutions*, Intern. Journ. of Foundations of Computer Science, Vol.2, No.3, 1991, pp. 207-220.

[C 71] S. Cook, *The complexity of theorem-proving procedures*, Proceedings, 3rd Annu. ACM Symp. on Theory of Computing, 1971, pp. 151-158.

[FGLSS 96] U. Feige, S. Goldwasser, L. Lovász, S. Safra, M. Szegedy, *Interactive Proofs and the Hardness of Approximating Cliques*, Journal of the ACM, Vol.43, No.2, March 1996, pp. 268-292.

[GKR 95] W.I. Garsarch, M.W. Krentel, K.J. Rappoport, *OptP as the Normal Behavior of NP-Complete Problems*, Math. Systems Theory 28, 1995, pp. 487-514.

[GJ 79] M. Garey, D. Johnson, *Computers and Intractability: A Guide to the Theory of $\mathcal{NP}$-Completeness*, Freeman, San Francisco, 1979.

[H 96] J. Håstad, *Clique is hard to approximate within $n^{1-\epsilon}$*, 28th Annu. ACM Symp. Theory of Computing, 1996, pp. 627-636.

[HW 97] H. Hempel, G. Wechsung, *The Operators min and max on the Polynomial Hierarchy*, Proc. 14th Symp. on Theoretical Aspects of Computer Science, 1997, pp. 93-104.

[JS 96] A. Jakoby, C. Schindelhauer, *On the Complexity of Worst Case and Expected Time in a Circuit*, Proc. 13th Symp. on Theoretical Aspects of Computer Science, 1996, pp. 195-306.

[K 88] M. Krentel, *The Complexity of Optimization Problems*, Journal of Computer and System Sciences 36, 1988, pp. 490-509.

[S 94] A. Selman, *A Taxonomy of Complexity Classes of Functions*, Journal of Computer and System Sciences 48, 1994, pp. 357-381.

[VW 95] H. Vollmer, K. Wagner, *Complexity Classes of Optimization Functions*, Information and Computation, 120 (2), 1995, pp. 198-219.

[W 88] K. Wagner, *Bounded Query Computations*, Proc. Structure in Complexity Theory, 1988, pp. 260-277.

[W 90] K. Wagner, *Bounded Query Classes*, SIAM Journal of Computing, Vol. 19, No. 5, 1990, pp 811-846.

Appendix

Theorem 5 MAX-1-GUESS *is* $\left(\mathcal{FP}_{tt}^{\mathcal{NP}} \cap \mathcal{UF}\right)$*-complete under* $\leq_{1\text{-}T}^{\mathrm{u}\mathcal{FP}}$*-reducibility.*

Proof: Let $\mathcal{O}$ be a polynomial time bounded nondeterministic Turing machine that verifies for a given NTM N, an input x, and a number t, whether max-1-guess$_N(x) \geq t$. Using $\mathcal{O}$ as an oracle, a TM M can compute max-1-guess$_N(x)$ by asking the oracle $q(|x|)$ unadaptive queries for a polynomial q.

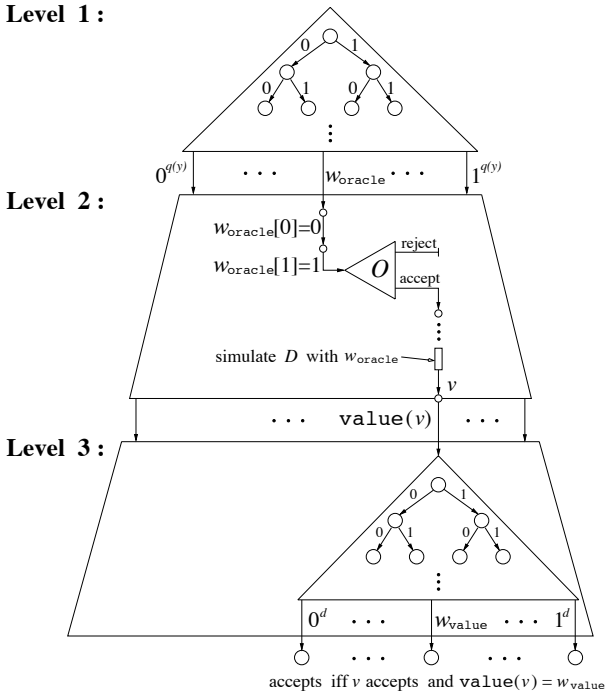

Figure 1: The computation tree of NTM N'.

Now we construct a nondeterministic TM N' within three steps by using the following technique, see figure 1. First N' chooses the answer string $w_{\mathtt{oracle}} \in \{0,1\}^{q(y)}$ of the oracle $\mathcal{O}$ such that these strings are lexicographically sorted at the leafs of the computation tree of N'. Further, N' verifies whether all binary digits $w_{\mathtt{oracle}}[i]$ of $w_{\mathtt{oracle}}$ with $w_{\mathtt{oracle}}[i] = 1$ corresponds to the positive answers of $\mathcal{O}$. At last N' simulates D, where the oracle queries are answered by the corresponding binary digits of $w_{\mathtt{oracle}}$. Define $w_{\mathtt{oracle}}(p)$ for a path p of the computation tree of N' as the answer string of $\mathcal{O}$. Let p be an accepting path of the computation tree of N'. Since p has an accepting leaf, only the digits $w_{\mathtt{oracle}}(p)[i] = 0$ might be wrong. For a path p let $\mathcal{T}(p) := \{\, j \in \{1,\ldots,q(y)\} \mid w_{\mathtt{oracle}}(p)[j] = 1 \,\}$. Since queries are non adaptive, for each pair p, p' of accepting paths with $\mathcal{T}(p) \neq \mathcal{T}(p')$ there exists an accepting path p'' with $\mathcal{T}(p'') = \mathcal{T}(p) \cup \mathcal{T}(p')$. Therefore there exists an accepting path p such that the set $\mathcal{T}(p)$ is maximal, i. e. for each path p' holds $\mathcal{T}(p') \subseteq \mathcal{T}(p)$. Note that p determines the output of D.

Now, N' can be modified to an admissible NTM, i.e. all paths from the starting state to a leaf of the computation tree T' have the same length d and N' has two nondeterministic choices at each state except the halting states.

Every leaf v of T' corresponds to a path p. Therefore it corresponds to the computation of D with oracle answers $w_{\mathtt{oracle}}(p)$, too. For an accepting leaf v of T' define value$(v) \in \mathbb{N}$ as the output of D with oracle answers $w_{\mathtt{oracle}}(p)$. For a rejecting leaf v let value$(v) := 0$.

Let N'' be a NTM that works like N' with the difference that N'' guesses nondeterministically a string $w_{\texttt{value}} \in \{0,1\}^d$ at a leaf v of T'. N'' accepts iff v accepts and $w_{\texttt{value}} = 0^{d-\text{value}(v)}1^{\text{value}(v)}$. Again we can assume, that N'' is an admissible NTM, that means the computation tree of N'' is a complete binary tree.

For a path p of N'' let $w_{\text{comp}}(p)$ denote the sequence of nondeterministic choices of N'' on p and define the subsequence $w_{\texttt{value}}(p) := w_{\text{comp}}(p)[d{+}1]..w_{\text{comp}}(p)[2d]$. Then for each pair p, p' of accepting paths of N'' with $w_{\texttt{oracle}}(p) = w_{\texttt{oracle}}(p')$ holds $w_{\texttt{value}}(p) = w_{\texttt{value}}(p')$.

Now, N''' is an admissible NTM that simulates N'' at first. Further, N''' makes $2d^2 + d$ and then $2d^3 + d^2 + d$ nondeterministic choices. N''' accepts on a computation path p iff

- $p[1] \dots p[d]$ indicates an accepting computation path of N' with leaf v,
- $p[1] \dots p[2d]$ indicates an accepting computation path of N'',
- $p[2d+1] \dots p[2d^2+3d]$ contains $(2d+1) \cdot \text{value}(v)$ right choices, and
- $p[2d^2+3d+1] \dots p[2d^3+3d^2+4d]$ contains $(2d^2+d+1) \cdot |\mathcal{T}(p[1] \dots p[d])|$ right choices. ∎