Optimization Problems with Approximation Schemes

Anders Malmström

Lehrgebiet Mathematische Grundlagen der Informatik
RWTH Aachen, Germany
E-mail: anders@informatik.rwth-aachen.de

Abstract. In this paper we extend recent work about the relationship between the syntactic description of NP optimization problems and their approximation properties. In contrast to Max SNP we consider problems that take arbitrary weighted structures as input instances and we use the framework of *Metafinite Model Theory* [5] to get a more general definability theory of optimization problems. We define a class Max Φ and show that every problem in this class has a *fully polynomial-time approximation scheme* (FPTAS), i.e., can be approximated to every desired accuracy ε in time polynomial in the size of the input and $1/\varepsilon$. An example for a problem in Max Φ is Knapsack.

1 Introduction

Many important optimization problems are NP-complete in their decision version. This rules out the possibility of finding in polynomial-time an optimal solution unless P=NP. However, with respect to approximability NP complete optimization problems may behave differently. Some admit efficient approximation algorithms, others like the Traveling Salesman Problem are as hard to approximate as to solve exactly (for a broad list of problems see [3]). What are the structural reasons for these differences?

A first break-through in this question was made when Papadimitriou and Yannakakis [12] discovered in some important cases a relationship between approximation properties of optimization problems and their representations by logical formulae. Motivated by Fagin's characterization of NP in terms of existential second-order logic on finite structures [4], they defined two classes of optimization problems, Max SNP and Max NP, and proved that for each problem $\mathcal{Q}$ in these classes there exists a polynomial-time algorithm that approximates the optimum with a constant relative error. Many natural maximization problems like Max Cut and Max Sat are contained in Max NP.

Building on this work Kolaitis and Thakur [8] investigated the logical definability of optimization problems in a systematic way. Let $\mathcal{Q}$ be an optimization problem whose input instances are finite structures $\mathfrak{A}$ of a fixed signature σ, and let $\psi(\bar{x}, \bar{S})$ be a first-order formula where $\bar{S}$ is a sequence of relation variables not contained in σ.

We say that ψ defines $\mathcal{Q}$ if the cost of an optimal solution of $\mathcal{Q}$ on input $\mathfrak{A}$ is definable by the expression

$$\text{opt}_{\mathcal{Q}}(\mathfrak{A}) = \max_{\bar{S}} |\{\bar{x} : \mathfrak{A} \models \psi(\bar{x}, \bar{S})\}|.$$

Recall that Σ_n (resp. Π_n) is the class of first-order formulae in prenex normal form with n alternating blocks of quantifiers beginning with $\exists$ (resp. $\forall$). Kolaitis and Thakur defined MAX Σ_n (resp. MAX Π_n) to be the class of all maximization problems definable by Σ_n-formulae (resp. Π_n-formulae).

Kolaitis and Thakur proved that these classes collapse to a strict hierarchy of four levels and that all maximization problems with polynomially bounded cost functions fall into this hierarchy. The classes MAX SNP and MAX NP are the closures of MAX Σ_0 and MAX Σ_1 under appropriate approximation-preserving reductions (so-called L-reductions). Thus, the class MAX Σ_1 provides a sufficient *syntactic criterion for approximability.*

Papadimitriou and Yannakakis also exhibited a number of complete problems for MAX SNP under L-reductions. The question whether these complete problems can be efficiently approximated with arbitrary accuracy has been answered in the negative. Arora et al. [1] proved that MAX SNP-hard problems cannot be approximated with arbitrarily small error (i.e., have PTAS), unless P=NP. Moreover Khanna et al. [7] showed that all constant approximable optimization problems can be L-reduced to a problem in MAX SNP.

Surprisingly the situation is not dual for minimization problems (for details we refer to [8] and [9]). However Kolaitis and Thakur [9] isolated two syntactic classes, called MIN $\text{F}^+\Pi_1$ and MIN $\text{F}^+\Pi_2(1)$ all whose problems are constant respectively log-approximable[1].

Many natural NP optimization problems take weighted structures as input instances, e.g., graphs with weighted vertices. Papadimitriou and Yannakakis [12] considered variants of MAX SNP and MAX NP that use weights in a rather limited way. They attached weights to the tuples of the input structure that can be substituted for the free variables $\bar{x}$ and showed that the approximation properties are provided. In a recent paper Zimand [13] analyzed the weighted variants of all syntactically defined classes that are known to have good approximation properties.

To extend this definability theory to a more general one of optimization problems with arbitrary weights we use the framework of *Metafinite Model Theory* (introduced by Grädel and Gurevich [5]), an extension of finite model theory. Metafinite structures are two-sorted structures that consist of a finite structure, an infinite structure and "weight"-functions from the finite to the infinite part. In this paper we only consider metafinite structures whose infinite part is a structure $\mathfrak{N}$ over the natural numbers, so-called *arithmetical structures.*

On arithmetical structures feasible solutions of NP optimization problems can be encoded by functions from the finite part into $\mathbb{N}$. All problems we consider

[1] A minimization problem is log-approximable if one can find a solution of cost at most $O(\log n)$ times larger than the cost of an optimal solution, where n is the size of the input.

have in common that the cost is a linear functional in the "solution"-function (this is the case for most optimization problems).

As in the unweighted case there is a connection between the syntactic description of an optimization problem and its approximation properties. All results of Papadimitriou/Yannakakis and Kolaitis/Thakur can be extended to arithmetical structures [10] but this is not subject to this paper.

We introduce a new class, MAX Φ, that contains only problems that admit a *fully polynomial-time approximation scheme* (FPTAS), i.e., problems that can be approximated to every desired accuracy ε in time polynomial in the size of the input structure and $1/\varepsilon$. MAX Φ consists of all problems that can be described in the following way:
On any given arithmetical input structure $\mathfrak{D}$

$$\mathrm{opt}_{\mathcal{Q}}(\mathfrak{D}) = \max_{f} \Big\{ L(\mathfrak{D}, f) : G(\mathfrak{D}, f) \leqslant B(\mathfrak{D}) \wedge (\mathfrak{D}, f) \models \psi \Big\},$$

where L is a linear functional, G and B are numerical invariants and ψ is a first-order formula that is monotone in f. Additionally we demand that G is monotone in a very homogeneous sense, we call it *h-monotone*. For instance linear functionals are h-monotone.

A typical example for MAX Φ is the KNAPSACK problem:
The MAX 0-1 KNAPSACK problem can be described as follows. Given are n objects $\{a_0, \ldots, a_{n-1}\}$, each with a weight $w(x)$ and a capacity $c(x)$, and an upper bound B. The goal is to choose a set of objects such that the total weight becomes maximal but the total capacity is less than B.

The input instance $\mathfrak{D} = (A, \mathfrak{N}, W)$ is an arithmetical structure where the primary part is the pure set $A = \{a_0, a_1, \ldots, a_{n-1}\}$, the secondary part is $\mathfrak{N}$ and W contains the two weight functions $w : A \to \mathbb{N}$ and $c : A \to \mathbb{N}$.
It is easy to see that MAX 0-1 KNAPSACK fits our description:

$$\begin{aligned} \mathrm{opt}_{\mathcal{Q}}(\mathfrak{D}) = \max_{f : A \to \mathbb{N}} \Big\{ \sum_{x \in A} f(x) w(x) : & \sum_{x \in A} f(x) c(x) \leqslant B \\ & \wedge (\mathfrak{D}, f) \models \forall x (f(x) \leqslant 1) \Big\}. \end{aligned}$$

It is well know that MAX 0-1 KNAPSACK admits an FPTAS (see [6,2] for details).

We prove the following result in section 3.

Theorem. *Every problem $\mathcal{Q}$ in* MAX Φ *admits an* FPTAS.

Thus we have a syntactic description of maximization problems that induces an FPTAS. Even more, the result provides a calculus for constructing problems with an FPTAS.

We can adapt this result for minimization problems as well.

2 Preliminaries

2.1 Arithmetical structures

Let us first fix some notation and abbreviations.
As in the first part Gothic letters $\mathfrak{A}, \mathfrak{D} \ldots$ stand for structures while the corresponding Latin letters $A, D \ldots$ denote their universes.

Let A be a finite set and let $S \subseteq A$ be a subset. We denote the characteristic function of S by $\chi_S : A \to \{0,1\}$. If $S = \{s\}$ is a singleton we write χ_s instead of $\chi_{\{s\}}$.

Let f and g be two functions with the same domain and range. We use the expression $f \sqsubset g$ to mean that $f(x) \leqslant g(x)$ for all x and $f \neq g$.

Definition 1. A *multiset* is a set where multiple occurrences of elements are allowed. A multiset M over a set R can be described by a function $m : R \to \mathbb{N}$ where $m(r)$ is the multiplicity of r in R. Multisets are denoted by double-brackets $\{\!\{\ldots\}\!\}$. Let $\mathrm{fm}(R)$ be the set of all finite multisets of R.
Multiset operations are operations $\Gamma : \mathrm{fm}(R) \to R$ mapping multisets to elements. Natural examples (e.g., for $R = \mathbb{N}, \mathbb{Q}, \mathbb{R}$) are sum, product, number of elements, median, maximum and minimum.

Definition 2. An *arithmetical structure* is a triple $\mathfrak{D} = (A, \mathfrak{N}, W)$ where

(i) the primary part A is a finite set.
(ii) $\mathfrak{N}$ is a structure over the natural numbers with multiset operations such that
 - $\mathfrak{N}$ has at least the two constants 0, 1, the functions $+, \cdot$, the multiset operations $\max, \min, \sum, \prod$ and the order relation $<$.
 - all functions, relations and operations of $\mathfrak{N}$ can be evaluated in polynomial-time.

(iii) W is a finite set of functions $w : A^k \to \mathbb{N}$.

The signature of $\mathfrak{D}$ is the pair $\sigma = (\sigma_N, \sigma_W)$ where each component is the set of symbols in the corresponding component.

Let us make precise what we mean by evaluation in polynomial-time. Functions have to be polynomial-time computable with respect to the binary representation. A relation is polynomial-time computable if its characteristic function is so. Let Γ be a multiset operation and let M be a finite multiset of natural numbers. We assume that the multiset is represented by listing all elements. Let $m_M(n)$ be the multiplicity of n in M. Then the cost of M is $\|M\| := \sum_n m_M(n) \log n$. Now Γ has to be polynomial-time computable with respect to this representation.

2.2 Optimization problems

Definition 3. An *NP optimization problem* is a quadruple
$\mathcal{Q} = (\mathcal{I}_Q, \mathcal{F}_Q, c_Q, \mathrm{opt})$ such that

- $\mathcal{I}_Q$ is the set of *input instances* of $\mathcal{Q}$. It is recognizable in polynomial-time.
- $\mathcal{F}_Q(I)$ is the set of *feasible solutions* for input $I \in \mathcal{I}_Q$. Here 'feasible' means that the size of the elements $S \in \mathcal{F}_Q(I)$ is polynomially bounded in the size of I and that the set $\{(I,S) : S \in \mathcal{F}_Q(I)\}$ is recognizable in polynomial-time.
- $c_Q : \{(I,S) : S \in \mathcal{F}_Q(I)\} \to \mathbb{N}$ is a polynomial-time computable function, called the *cost function*.
- $\mathrm{opt} \in \{\max, \min\}$.

The cost of an optimal solution on input instance I is

$$\mathrm{opt}_Q(I) = \mathrm{opt}_{S\in\mathcal{F}_Q(I)} c_Q(I,S).$$

The name NP optimization problem is justified since this definition induces that for every problem $\mathcal{Q}$, the associated decision problem is in NP: Given an instance I of $\mathcal{Q}$ and a natural number k, is there a solution $S \in \mathcal{F}_Q(I)$ such that $c_Q(I,S) \geqslant k$ when opt = max, (or $c_Q(I,S) \leqslant k$, when opt = min).

Approximation. Let $\mathcal{Q}$ be an NP optimization problem and let A be an algorithm which returns to each input instance $I \in \mathcal{I}_Q$ a feasible solution $A(I) \in \mathcal{F}_Q(I)$. We call A an ε-approximation algorithm if the relative error of A is smaller than ε for all $I \in \mathcal{I}_Q$, i.e.,

$$\frac{|c_Q(I,A(I)) - \mathrm{opt}_Q(I)|}{\max\{\mathrm{opt}_Q(I), c_Q(I,A(I))\}} \leqslant \varepsilon.$$

Definition 4. We say that an optimization problem $\mathcal{Q}$ has a *polynomial-time approximation scheme* (PTAS) if there exists an algorithm A which returns for each rational $\varepsilon > 0$ and input instance I a solution with a relative error less than ε in time polynomial in $|I|$ (depending on ε in an arbitrary way).

A PTAS is called a *fully polynomial-time approximation scheme* (FPTAS)[2] if the time bound for A is polynomial in $|I|$ and $1/\varepsilon$.

2.3 Numerical invariants

Let $\mathfrak{C}$ be a class of arithmetical structures of fixed signature σ.

Definition 5. A *numerical invariant* on $\mathfrak{C}$ is a function I that assigns to every structure $\mathfrak{D} = (A, \mathfrak{N}, W)$ a natural number such that isomorphisms between structures are preserved, i.e., for every isomorphism $h : \mathfrak{D} \to \mathfrak{D}' : I(\mathfrak{D}) = I(\mathfrak{D}')$. Thus numerical invariants map to each isomorphism class a numerical value.

Complexity. Let $\mathfrak{D} = (A, \mathfrak{N}, W)$ be an arithmetical structure and let

$\max \mathfrak{D} := \max\limits_{w\in W} \max\limits_{\bar{a}} \|w(\bar{a})\|$ be the cost of the maximal weight in $\mathfrak{D}$.

We say that a numerical invariant is polynomial-time computable if it can be computed in time less than $p(|A|, \max \mathfrak{D})$ where $p(n,m)$ is a polynomial.

[2] Sometimes we use FPTAS to denote the class of all problems with an FPTAS.

A special kind of numerical invariants are functionals:
Let f be a new m-ary function symbol not in σ_N. For each $\mathfrak{D} \in \mathfrak{C}$, let $(\mathfrak{D}, f)$ be the expansion of $\mathfrak{D}$ by a new weight function f. Denote the class of all such expansions by $\mathfrak{C}(f)$. Let $F : \mathfrak{C}(f) \to \mathbb{N}$ be a numerical invariant. Once a structure $\mathfrak{D}$ is fixed we can view F as a *functional* assigning to every (interpretation of) f the value $F(\mathfrak{D}, f)$.

Definition 6. Let f be a m-ary function symbol and let $G : \mathfrak{C}(f) \to \mathbb{N}$ be a numerical invariant. We say that G is *h-monotone in* f (h stands for homogeneous) if for every fixed $\mathfrak{D} \in \mathfrak{C}$ the following two conditions are satisfied.

(i) $f \sqsubset g \Longrightarrow G(\mathfrak{D}, f) < G(\mathfrak{D}, g)$.
(ii) $G(\mathfrak{D}, f) \leqslant G(\mathfrak{D}, g) \Longrightarrow G(\mathfrak{D}, f + h) \leqslant G(\mathfrak{D}, g + h)$, for all functions f, g, h from A^m to the natural numbers $\mathbb{N}$.

Natural examples for h-monotone invariants in one function are linear functionals[3]. But the class of h-monotone invariants is strictly larger than the class of linear functionals. To see this, observe that for an arbitrary strongly monotone increasing function $M : \mathbb{N} \to \mathbb{N}$, if $G(\mathfrak{D}, f)$ is h-monotone in f, then so is $M \circ G(\mathfrak{D}, f)$.
Or, let $\mathfrak{C}$ be the class of all finite graphs with weighted vertices, $G = (V, E)$, $w : V \to \mathbb{N}^+$, and let f be unary. Then the invariant

$$\begin{aligned} F : \mathfrak{C}(f) &\to \mathbb{N} \\ (G, w, f) &\mapsto \prod_{v \in V} w(v)^{f(v)} \text{ is h-monotone in } f. \end{aligned}$$

Proposition 7. *Let G be a h-monotone numerical invariant and let e, f, g, h be arbitrary functions from A^m to $\mathbb{N}$ and let λ be a natural number. Then*

1. $G(\mathfrak{D}, e) \leqslant G(\mathfrak{D}, f) \wedge G(\mathfrak{D}, g) \leqslant G(\mathfrak{D}, h) \Longrightarrow G(\mathfrak{D}, e + g) \leqslant G(\mathfrak{D}, f + h)$.
2. $G(\mathfrak{D}, f) \leqslant G(\mathfrak{D}, g) \Longrightarrow G(\mathfrak{D}, \lambda \cdot f) \leqslant G(\mathfrak{D}, \lambda \cdot g)$.

Proof. It is clear that 1. implies 2. By Definition 6 we get that

$$G(\mathfrak{D}, g) \leqslant G(\mathfrak{D}, h) \Longrightarrow G(\mathfrak{D}, e + g) \leqslant G(\mathfrak{D}, e + h) \text{ and that}$$
$$G(\mathfrak{D}, e) \leqslant G(\mathfrak{D}, f) \Longrightarrow G(\mathfrak{D}, e + h) \leqslant G(\mathfrak{D}, f + h).$$

The combination of the two inequalities proves 1. □

2.4 Optimization on arithmetical structures

Let $\mathcal{Q}$ be an NP optimization problem. Every input instance of $\mathcal{Q}$ can be represented as an arithmetical structure of some fixed signature. Since the size of feasible solutions of $\mathcal{Q}$ is polynomially bounded in $\max \mathfrak{D}$ and $|A|$ (compare Definition 3) they can be encoded as tuples of weight functions.

[3] A functional L is *linear* if $L(\mathfrak{D}, f + g) = L(\mathfrak{D}, f) + L(\mathfrak{D}, g)$ for all functions f and g.

Let $\sigma = (\sigma_N, \sigma_W)$ be the signature of $\mathcal{Q}$. Define $\mathfrak{C}$ to be the class of σ-structures representing input instances of $\mathcal{Q}$ and let $\bar{f} = f_1, \dots, f_k$ be a tuple of new weight function symbols not in σ_W. Denote the class of all expansions by $\mathfrak{C}(\bar{f})$. For fixed $\mathfrak{D}$ let $\mathcal{S}(\mathfrak{D})$ be the set of all feasible solution functions $\bar{f}$. Note that $\mathcal{S}(\mathfrak{D})$ is recognizable in polynomial-time. The cost function F of $\mathcal{Q}$ is a numerical invariant on $\mathfrak{C}(\bar{f})$:

$$F : \mathfrak{C}(\bar{f}) \to \mathbb{N}.$$

Thus the optimization problem $\mathcal{Q}$ can be described by

$$\mathrm{opt}_{\mathcal{Q}}(\mathfrak{D}) = \mathrm{opt}\{F(\mathfrak{D}, \bar{f}) : \bar{f} \in \mathcal{S}(\mathfrak{D})\},$$

where $\mathrm{opt} \in \{\max, \min\}$.

The set $\mathcal{S}(\mathfrak{D})$ will often be described by a logical formula $\psi(\bar{f})$, i.e.,

$$\bar{f} \in \mathcal{S}(\mathfrak{D}) \iff \mathfrak{D} \models \psi(\bar{f}).$$

The only condition for ψ is that it can be evaluated in polynomial-time.

3 The class MAX Φ

Let $\sigma = (\sigma_N, \sigma_W)$ be a signature of arithmetical structures.

Definition 8. The class MAX Φ consists of all maximization problems $\mathcal{Q}$ whose input instances are arithmetical structures $\mathfrak{D} = (A, \mathfrak{N}, W)$ over σ such that

$$\mathrm{opt}_{\mathcal{Q}}(\mathfrak{D}) = \max_f \left\{ L(\mathfrak{D}, f) : G(\mathfrak{D}, f) \leqslant B(\mathfrak{D}) \wedge (\mathfrak{D}, f) \models \psi \right\}, \text{ where}$$

(i) $f : A^m \to \mathbb{N}$, $m \geqslant 1$,
(ii) $L(\mathfrak{D}, f)$ is a linear functional,
(iii) $G(\mathfrak{D}, f)$ is a numerical invariant which is h-monotone in f,
(iv) $B(\mathfrak{D})$ is a numerical invariant which only depends on $\mathfrak{D}$ and not on f,
(v) $\psi = \forall \bar{x} \varphi(\bar{x}, f(\bar{x}))$, where $\bar{x}$ is an m-tuple, $\varphi(\bar{x}, y)$ is a σ-formula that is monotone in y,
i.e., if $\mathfrak{D} \models \varphi(\bar{a}, u)$ for some $\bar{a} \in A^m$ and $u \in \mathbb{N}$ then $\mathfrak{D} \models \varphi(\bar{a}, v)$ for all $v < u$.
Consequently $f' \sqsubset f$ and $(\mathfrak{D}, f) \models \psi$ imply that $(\mathfrak{D}, f') \models \psi$.
(vi) L, G and B are computable in polynomial-time and ψ is evaluable in polynomial-time.

If $\mathcal{Q}$ has a feasible solution for $\mathfrak{D}$ then clearly $(\mathfrak{D}, 0) \models \psi$.

Remark: Note that by (i) of Definition 6 (h-monotonicity) $f(a) \leqslant B(\mathfrak{D})$ for all $a \in A$. This implies that only finitely many functions satisfy the constraints (i) to (vi) and thus if $\mathcal{Q}$ has at least one feasible solution for $\mathfrak{D}$ the maximum exists.

Example 9. The MAX KNAPSACK problem is the same as MAX 0-1 KNAPSACK, except that any object can be used arbitrarily often.

We can encode the input instance as an arithmetical structure $\mathfrak{D} = (A, \mathfrak{N}, W)$ in the following way. The primary part is the pure set $A = \{a_0, a_1, \dots, a_{n-1}\}$, the secondary part is $\mathfrak{N}$ and W contains the two weight functions $w : A \to \mathbb{N}$ and $c : A \to \mathbb{N}$.

$$\mathrm{opt}_{\mathcal{Q}}(\mathfrak{D}) = \max_{f:A\to\mathbb{N}} \Big\{ \sum_{x\in A} f(x)w(x) : \sum_{x\in A} f(x)c(x) \leqslant B \Big\}.$$

As already mentioned it is well know that MAX KNAPSACK admits an FPTAS [6]. We can now formulate our main result.

Theorem 10. *Every problem in* MAX Φ *has an* FPTAS.

To prove this theorem we use two different algorithms. The first one gives an exact solution but its running time is not polynomial, it is a so-called pseudo-polynomial algorithm. The second one is a polynomial-time approximation algorithm with an error that depends on the maximal weight and n. We then combine the two algorithms and get the desired result.

Consider the following algorithm which is a generalization of an algorithm for the KNAPSACK problem in [11].

Algorithm `ExSol`:
Let $\mathcal{Q}$ be an optimization problem in MAX Φ and let $\mathfrak{D} = (A, \mathfrak{N}, W)$ be an input instance of $\mathcal{Q}$. W.l.o.g. let $m = 1$ and $A = \{a_0, \dots, a_{n-1}\}$. `ExSol` is a dynamic programming algorithm that constructs an ascending chain $M_0, \dots, M_n$ of sets of feasible solutions $f : A \to \mathbb{N}$.

1. $M_0 := \{0\}$.
2. For $i = 0, \dots, n-1$ <u>do</u>
   (a) $M_{i+1} := M_i$.
   (b) For each $f \in M_i$ <u>do</u>
   $k := 1$
   while $G(\mathfrak{D}, f + k \cdot \chi_{a_i}) \leqslant B(\mathfrak{D}) \wedge (\mathfrak{D}, f + k \cdot \chi_{a_i}) \models \psi$ <u>do</u>
   add $(f + k \cdot \chi_{a_i})$ to M_{i+1}
   $k := k + 1$
   <u>od</u>
   <u>od</u>
   (c) If f and g belong to M_{i+1} and $L(\mathfrak{D}, f) = L(\mathfrak{D}, g)$, delete g if $G(\mathfrak{D}, g) \geqslant G(\mathfrak{D}, f)$ else delete f.
   <u>od</u>
3. As optimal solution take the $f \in M_n$ with maximal value $L(\mathfrak{D}, f)$.

Remember that $(f + k \cdot \chi_{a_i})(x) = \begin{cases} f(x) & : x \neq a_i \\ f(x) + k & : x = a_i \end{cases}$

Note that the while loop terminates since for feasible f, $f(a_i) \leqslant B(\mathfrak{D})$.

The next lemma gives some properties of the algorithm `ExSol`.

Lemma 11. *Let $\mathcal{Q}$ be an optimization problem in* $\text{MAX}\ \Phi$ *and $\mathfrak{D}$ an input instance. If $f \in M_i$ after executing algorithm* `ExSol`*, then*

1. $f(a_j) = 0 \quad \forall j \geqslant i$.
2. $G(\mathfrak{D}, f) \leqslant B(\mathfrak{D})$.
3. $(\mathfrak{D}, f) \models \psi$.
4. *If $g \in M_i$ and $L(\mathfrak{D}, f) = L(\mathfrak{D}, g)$ then $g \equiv f$.*
5. *Let $g : A \to \mathbb{N}$ with $g(a_j) = 0 \quad \forall j \geqslant i$, $G(\mathfrak{D}, g) \leqslant B(\mathfrak{D})$ and $(\mathfrak{D}, g) \models \psi$ then there exists some $h \in M_i$ with $L(\mathfrak{D}, h) = L(\mathfrak{D}, g)$.*
6. *Let $g : A \to \mathbb{N}$ with $g(a_j) = 0 \quad \forall j \geqslant i$, $(\mathfrak{D}, g) \models \psi$ and $L(\mathfrak{D}, g) = L(\mathfrak{D}, f)$ then $G(\mathfrak{D}, g) \geqslant G(\mathfrak{D}, f)$.*

Proof. Points (1) to (4) are straightforward.
To prove (5) assume that $i > 0$ (if $i = 0$ then (5) clearly holds). We argue by induction on $\ell = \max\{j : g(a_j) \neq 0\}$. If $\ell = 0$ then (5) holds trivially. Assume $\ell \neq 0$. By induction hypothesis there is an element $h \in M_\ell$ with $L(\mathfrak{D}, h) = L(\mathfrak{D}, \bar{g})$,

$$\text{where } \bar{g}(x) = \begin{cases} g(x) & : x \neq a_\ell \\ 0 & : x = a_\ell \end{cases}$$

By step (2a) of algorithm `ExSol`, the function $\tilde{h}$

$$\text{with} \quad \tilde{h}(x) = \begin{cases} h(x) & : x \neq a_\ell \\ g(a_\ell) & : x = a_\ell \end{cases}$$

has been included in $M_{\ell+1}$, Because $\tilde{h}(a_\ell) = g(a_\ell)$ it is clear that $(\mathfrak{D}, \tilde{h}) \models \psi$. Then either this function itself or an element $\tilde{h}'$ with $L(\mathfrak{D}, \tilde{h}) = L(\mathfrak{D}, \tilde{h}')$ is in M_i. Hence (5) holds.

The proof of (6) is by induction on i. (6) trivially holds for $i = 0$. Assume that (6) holds for $i - 1$, $i > 0$.

Define

$$\bar{g}(x) := \begin{cases} g(x) & : x \neq a_{i-1} \\ 0 & : x = a_{i-1} \end{cases}$$

By (5) there exists $\bar{h} \in M_{i-1}$ with $L(\mathfrak{D}, \bar{h}) = L(\mathfrak{D}, \bar{g})$. By the induction hypothesis $G(\mathfrak{D}, \bar{g}) \geqslant G(\mathfrak{D}, \bar{h})$. The h-monotonicity of G implies that $G(\mathfrak{D}, g) \geqslant G(\mathfrak{D}, h)$,

$$\text{where } h(x) = \begin{cases} \bar{h}(x) & : x \neq a_{i-1} \\ g(a_{i-1}) & : x = a_{i-1} \end{cases}$$

Because L is a linear functional $L(\mathfrak{D}, h) = L(\mathfrak{D}, g)$ and because $\bar{h} \in M_{i-1}$ and $h(a_{i-1}) = g(a_{i-1})$ it is clear that $(\mathfrak{D}, h) \models \psi$ since $(\mathfrak{D}, g) \models \psi$. Finally note that $G(\mathfrak{D}, h) \geqslant G(\mathfrak{D}, f)$ since otherwise f would have been deleted in step (2c). $\square$

Theorem 12. *Algorithm* `ExSol` *solves* $\mathcal{Q}$ *correctly in*

$$O(n \cdot (opt_{\mathcal{Q}}(\mathfrak{D}))^3) \text{ steps.}$$

Proof. The correctness of `ExSol` is an immediate consequence of Lemma 11. Note that the running time of `ExSol` need not be bounded by a polynomial. The cardinality of each M_i is smaller than $\mathrm{opt}_{\mathcal{Q}}(\mathfrak{D})$ because there is at most one function with the same cost which is not larger than the optimum. For each $f \in M_i$ the update process needs at most $(\mathrm{opt}_{\mathcal{Q}}(\mathfrak{D}))^2$ steps. This is since clearly $f'(a_i) \leqslant \mathrm{opt}_{\mathcal{Q}}(\mathfrak{D})$ for all feasible extensions f' of f and step (2c) can be done simultaneously using at most $\mathrm{opt}_{\mathcal{Q}}(\mathfrak{D})$ time-steps. Thus the update from M_i to M_{i+1} takes less than $O((\mathrm{opt}_{\mathcal{Q}}(\mathfrak{D}))^3)$ steps. Note that each step can be done in polynomial-time. □

Let $\mathcal{Q} \in \textsc{Max}\ \Phi$ and let us fix an input instance $\mathfrak{D}$. Because L is a linear functional it is clear that $L(\mathfrak{D}, f) = \sum_a f(a) \cdot L(\mathfrak{D}, \chi_a)$. Thus $L(\mathfrak{D}, \chi_a)$ is a weight term and is polynomial-time computable. Define the function

$$\begin{aligned} u : A \to \mathbb{N} \ \text{ such that } & G(\mathfrak{D}, u(a) \cdot \chi_a) \leqslant B(\mathfrak{D}), \text{ and} \\ & G(\mathfrak{D}, (u(a)+1) \cdot \chi_a) > B(\mathfrak{D}) \text{ for all } a \in A. \end{aligned}$$

The function u is polynomial-time computable and is an upper bound for f. Clearly $\mathrm{opt}_{\mathcal{Q}}(\mathfrak{D}) \leqslant n \cdot \max u(x) \cdot \max L(\chi_a)$.

Lemma 13. *Let f be a function such that $G(\mathfrak{D}, f) \leqslant B(\mathfrak{D})$. Then*

$$\sum_{x \in A} \frac{f(x)}{u(x)+1} \leqslant 1.$$

Proof. Let $b \in A$ such that $G(\mathfrak{D}, (u(a)+1)\chi_a) \geqslant G(\mathfrak{D}, (u(b)+1)\chi_b)$ for all $a \in A$.
Define the function

$$g(y) = \frac{f(y) \cdot (u(b)+1)}{(u(y)+1)}$$

and let $\lambda = \prod_{x \in A} (u(x)+1)$. It is clear that for all fixed $a \in A$

$$G(\mathfrak{D}, \lambda \cdot g(a) \cdot f(a)(u(a)+1)\chi_a) \geqslant G(\mathfrak{D}, \lambda \cdot g(a) \cdot f(a)(u(b)+1)\chi_b),$$

and since $g(a) \cdot (u(a)+1) = f(a) \cdot (u(b)+1) \in \mathbb{N}$ we can use the second point of Proposition 7 and we get

$$G(\mathfrak{D}, \lambda \cdot f(a) \cdot \chi_a) \geqslant G(\mathfrak{D}, \lambda \cdot g(a) \cdot \chi_b).$$

The first point of Proposition 7 allows us to take the sum over all a:

$$G(\mathfrak{D}, \lambda \cdot \sum_{a \in A} f(a)\chi_a) \geqslant G(\mathfrak{D}, \lambda \cdot \sum_{a \in A} g(a) \cdot \chi_b).$$

On the other hand by definition:

$$G(\mathfrak{D}, \lambda \cdot \sum_{a \in A} f(a)\chi_a) \leqslant G(\mathfrak{D}, \lambda \cdot (u(b)+1)\chi_b).$$

This implies that $\lambda \cdot (u(b)+1) \geqslant \lambda \cdot \sum_{a \in A} g(a)$, and it is straightforward that

$$1 \geqslant \sum_{a \in A} \frac{g(a)}{u(b)+1} = \sum_{a \in A} \frac{f(a)}{u(a)+1}.$$

□

The next theorem shows that computing the function u directly leads to an approximation with an error less than the maximal weight $L(\mathfrak{D}, \chi_a)$, provided f is not bounded by ψ, i.e., $(\mathfrak{D}, f) \models \psi$ for all functions f.

Theorem 14. *Let $d \in A$ be such that*

$$(u(a)+1)L(\mathfrak{D}, \chi_a) \leqslant (u(d)+1)L(\mathfrak{D}, \chi_d) \text{ for all } a \in A.$$

Then

$$u(d)L(\mathfrak{D}, \chi_d) \leqslant opt_{\mathcal{Q}}(\mathfrak{D}) \leqslant (u(d)+1)L(\mathfrak{D}, \chi_d).$$

Proof. The left inequality holds by the definition of u. Let f be an optimal solution of $\mathcal{Q}$. By assumption we get for each fixed $a \in A$ that

$$f(a) \cdot L(\mathfrak{D}, \chi_a) \leqslant (u(d)+1)L(\mathfrak{D}, \chi_d)\frac{f(a)}{u(a)+1}.$$

If we take the sum over all $a \in A$ then

$$\sum_{a \in A} f(a) \cdot L(\mathfrak{D}, \chi_a) \leqslant (u(d)+1)L(\mathfrak{D}, \chi_d) \sum_{a \in A} \frac{f(a)}{u(a)+1}.$$

Because $G(\mathfrak{D}, f) \leqslant B(\mathfrak{D})$ it follows by Lemma 13 that

$$\mathrm{opt}_{\mathcal{Q}}(\mathfrak{D}) = \sum_{a \in A} f(a) \cdot L(\mathfrak{D}, \chi_a) \leqslant (u(d)+1)L(\mathfrak{D}, \chi_d).$$

□

We now discuss the case where f is bounded by ψ.
Define the function

$$s : A \to \mathbb{N} \ \text{ such that } (\mathfrak{D}, s) \models \psi.$$
$$(\mathfrak{D}, s + \chi_a) \not\models \psi \text{ for all } a \in A.$$

Observe that this function is polynomial-time computable too. Without loss of generality we can assume that $s \sqsubseteq u$. As in the unbounded case we show that computing s leads to an approximate solution $\hat{f}$.

Initially define $\hat{f} = 0$ and define the functions $u_1 := u$ and $s_1 := s$. Consider the unbounded problem and compute $d \in A$ as in Theorem 14. Define

$$d_1 := d \text{ and } \hat{f}(d_1) := s(d_1).$$

Let $\mathcal{Q}_1$ be the subproblem of $\mathcal{Q}$ where the function is predefined on d_1.
Thus $\mathcal{Q}_1$ looks for an $f \sqsubseteq s$, $f(d_1) = 0$, such that $L(\mathfrak{D}, f)$ becomes maximal and $G(\mathfrak{D}, f + \hat{f}(d_1)\chi_{d_1}) \leqslant B(\mathfrak{D})$.
To approximate problem $\mathcal{Q}_1$ recompute the function u:

$$u_2 : A \setminus \{d_1\} \to \mathbb{N} \text{ and set } s_2 := \min(u_2, s).$$

Again consider the unbounded version of $\mathcal{Q}_1$ to compute the $d \in A \setminus \{d_1\}$ as in Theorem 14. Define

$$d_2 := d \text{ and } \hat{f}(d_2) := s_2(d_2).$$

Iterate this process until $\hat{f}$ is uniquely defined. Notice that this procedure is of polynomial length. Each step is polynomial-time and there are at most n steps.

As a by product this algorithm gives an order on A, the sequence $d_1, \ldots, d_n$.

Lemma 15. *Fix $i \in \{1, \ldots, n\}$ and $b \leqslant u_i(d_i)$. Let h be a function such that*

$$G(\mathfrak{D}, h) \leqslant G(\mathfrak{D}, b \cdot \chi_{d_i}) \text{ and } h(d_j) = 0 \text{ for } j < i.$$

Then

$$L(\mathfrak{D}, h) \leqslant L(\mathfrak{D}, (b+1)\chi_{d_i}).$$

Proof. This is a straightforward application of Theorem 14.
Since G is h-monotone

$$G(\mathfrak{D}, h + (u_i(d_i) - b)\chi_{d_i}) \leqslant G(\mathfrak{D}, u_i(d_i)\chi_{d_i})$$

and thus by Theorem 14

$$L(\mathfrak{D}, h + (u_i(d_i) - b)\chi_{d_i}) \leqslant (u_i(d_i) + 1)L(\mathfrak{D}, \chi_{d_i}).$$

Because L is a linear functional the statement follows. □

Theorem 16. *Let $\hat{f}$ be constructed as above. Then*

$$opt_{\mathcal{Q}}(\mathfrak{D}) \leqslant L(\mathfrak{D}, \hat{f}) + (2n+2) \max L(\mathfrak{D}, \chi_a).$$

Proof. Suppose that f^* is an optimal solution and $d_1, \ldots, d_n$ is the sequence as above. Let ℓ be the natural number such that

$$\hat{f}(d_i) = s(d_i) \text{ for } i = 1, \ldots, \ell - 1 \text{ and } \hat{f}(d_\ell) < s(d_\ell).$$

Clearly $\ell \leqslant n$ if $\hat{f} \neq f^*$.

We now construct a sequence of functions such that

1. $0 = f'_0 \sqsubseteq f_1 \sqsubseteq f'_1 \sqsubseteq f_2 \sqsubseteq \ldots \sqsubseteq f_{n+1} = f^*$
2. $f'_i = f_i + \chi_k$ for some $k \geqslant i$ if $\hat{f}(d_i) > 0$
3. $f'_i = f_i$ if $\hat{f}(d_i) = 0$
4. $G(\mathfrak{D}, f_i) \leqslant G(\mathfrak{D}, \sum_{j=1}^{i} \hat{f}(d_j)\chi_{d_j}), i \leqslant n$
5. $G(\mathfrak{D}, f'_i) > G(\mathfrak{D}, \sum_{j=1}^{i} \hat{f}(d_j)\chi_{d_j}), i \leqslant n$ if $\hat{f}(d_i) > 0$

We get these functions just by "filling up" along $d_1, d_2, \ldots, d_n$.
More precisely, f_i is the maximal function that satisfies point 4 and there exists a $j_0 \geqslant i$ such that

$$f_i(d_j) = \begin{cases} f^*(d_j) & \text{if } j < j_0, \\ 0 & \text{if } j > j_0. \end{cases}$$

Claim A: Suppose that $i < \ell$ then

$$L(\mathfrak{D}, f_{i+1} - f'_i) \leqslant L(\mathfrak{D}, (\hat{f}(d_{i+1}) + 1)\chi_{d_{i+1}}).$$

Since $i \leqslant \ell - 1$, it follows that $f_i(d_i) = f^*(d_i)$ and thus $(f_{i+1} - f'_i)(d_j) = 0$ for $j \leqslant i$. Now we argue that

$$G(\mathfrak{D}, f_{i+1} - f'_i) \leqslant G(\mathfrak{D}, \hat{f}(d_{i+1})\chi_{d_{i+1}}).$$

Assume the contrary:

$$G(\mathfrak{D}, f_{i+1} - f'_i) > G(\mathfrak{D}, \hat{f}(d_{i+1})\chi_{i+1}).$$

This implies that

$$\begin{aligned} G(\mathfrak{D}, f_{i+1}) &> G(\mathfrak{D}, f'_i + \hat{f}(d_{i+1})\chi_{d_{i+1}}) && \text{since } G \text{ h-monotone} \\ &\geqslant G(\mathfrak{D}, \sum_{j=1}^{i+1} \hat{f}(d_j)\chi_{d_j}) && \text{by point 5} \end{aligned}$$

But this contradicts point 4. Claim A now follows by Lemma 15.

Claim B: $L(\mathfrak{D}, f^* - f'_\ell) \leqslant L(\mathfrak{D}, 2 \cdot \chi_{d_\ell})$.

To prove this we have to show the assumptions of Lemma 15 with $b = 1$.
By definition, $(f^* - f'_\ell)(d_j) = 0$ for $j < \ell$.
For the second assumption assume the contrary:

$$G(\mathfrak{D}, f^* - f'_\ell) > G(\mathfrak{D}, \chi_{d_\ell}).$$

This implies that

$$G(\mathfrak{D}, f^*) > G(\mathfrak{D}, \sum_{j \leqslant \ell} \hat{f}_j(e_j)\chi_{d_j} + \chi_{d_\ell}) \qquad \text{by point 5}$$

But $G(\mathfrak{D}, \sum_{j \leqslant \ell} \hat{f}_j(d_j)\chi_{d_j} + \chi_{d_\ell}) > B(\mathfrak{D})$ because $\hat{f}_i(d_\ell) < s(d_\ell)$ is maximal.

Claim C: $L(\mathfrak{D}, f'_i - f_i) \leqslant \max L(\mathfrak{D}, \chi_a)$.

These tree claims prove the theorem:

$$\begin{aligned}
\mathrm{opt}_{\mathcal{Q}}(\mathfrak{D}) =& L(\mathfrak{D}, f^*) \\
=& \sum_{i=0}^{\ell-1} L(\mathfrak{D}, f_{i+1} - f'_i) + \sum_{i=1}^{\ell} L(\mathfrak{D}, f'_i - f_i) + L(\mathfrak{D}, f^* - f'_\ell) \\
\leqslant& \sum_{i=0}^{\ell-1} L(\mathfrak{D}, (\hat{f}(d_{i+1}) + 1)\chi_{d_{i+1}}) + n \cdot \max L(\mathfrak{D}, \chi_a) \\
&+ 2 \cdot L(\mathfrak{D}, \chi_{d_\ell}) \\
\leqslant& L(\mathfrak{D}, \hat{f}) + (2n+2) \max L(\mathfrak{D}, \chi_a)
\end{aligned}$$

□

Proof (of Theorem 10). Let $\mathcal{Q} \in \text{MAX } \Phi$ and $\mathfrak{D}$ be a input instance. We first consider the case where f is unbounded. We have seen that algorithm `ExSol` solves $\mathcal{Q}$ exactly but it is not a polynomial-time algorithm.

One way to make `ExSol` more efficient is the so-called *scaling technique*. We assign to the last bit in the binary representation of the weights $L(\mathfrak{D}, \chi_a)$ a zero, i.e., if $L(\mathfrak{D}, \chi_a) = 15$ (or 1111 in binary) then $L'(\mathfrak{D}, \chi_a) = 14$ (or 1110). This does not give too large an error as we will see.

For all $a \in A$ let $L'(\mathfrak{D}, \chi_a) = 2 \cdot \lfloor L(\mathfrak{D}, \chi_a)/2 \rfloor$ and let f^* and f' be optimal solutions of $\mathcal{Q}$ and $\mathcal{Q}'$ respectively, where $\mathcal{Q}'$ is obtained from $\mathcal{Q}$ by replacing L by L'. It is clear that $f^* \sqsubseteq u$. Then

$$\begin{aligned}
\sum_{a \in A} f^*(a) L(\mathfrak{D}, \chi_a) &\geqslant \sum_{a \in A} f'(a) L(\mathfrak{D}, \chi_a) \geqslant \sum_{a \in A} f'(a) L'(\mathfrak{D}, \chi_a) \\
&\geqslant \sum_{a \in A} f^*(a) L'(\mathfrak{D}, \chi_a) \geqslant \sum_{a \in A} f^*(a)(L(\mathfrak{D}, \chi_a) - 2) \\
&\geqslant \sum_{a \in A} f^*(a) L(\mathfrak{D}, \chi_a) - 2 \cdot \sum_{a \in A} u(a)
\end{aligned}$$

Thus

$$\begin{aligned}
\mathrm{opt}_{\mathcal{Q}}(\mathfrak{D}) - \mathrm{opt}_{\mathcal{Q}'}(\mathfrak{D}) &= \sum_{a \in A} f^*(a) L(\mathfrak{D}, \chi_a) - \sum_{a \in A} f'(a) L'(\mathfrak{D}, \chi_a) \\
&\leqslant 2 \cdot \sum_{a \in A} u(a)
\end{aligned}$$

In general if we ignore the last t bits we get

$$\mathrm{opt}_{\mathcal{Q}}(\mathfrak{D}) - \mathrm{opt}_{\mathcal{Q}'}(\mathfrak{D}) \leqslant 2^t \cdot \sum_{a \in A} u(a)$$

It is now easy to see that $\mathrm{opt}_{\mathcal{Q}'}(\mathfrak{D})$ is an ε-approximation of $\mathcal{Q}$ with $\varepsilon = 2^t \cdot I(\mathfrak{D})$, where

$$I(\mathfrak{D}) = \frac{\sum u(a)}{\max_i\{u(i) \cdot L(\mathfrak{D}, \chi_i)\}}.$$

This is because

$$\begin{aligned}
\frac{\mathrm{opt}_{\mathcal{Q}}(\mathfrak{D}) - \mathrm{opt}_{\mathcal{Q}'}(\mathfrak{D})}{\mathrm{opt}_{\mathcal{Q}}(\mathfrak{D})} &= \frac{L(\mathfrak{D}, f^*) - L'(\mathfrak{D}, f')}{L(\mathfrak{D}, f^*)} \\
&= \frac{\sum f^*(a) L(\mathfrak{D}, \chi_a) - \sum f'(a) L'(\mathfrak{D}, \chi_a)}{\sum f^*(a) L(\mathfrak{D}, \chi_a)} \\
&\leqslant \frac{2^t \cdot \sum u(a)}{\max_i\{u(i) \cdot L(\mathfrak{D}, \chi_i)\}} = \; 2^t \cdot I(\mathfrak{D})
\end{aligned}$$

$I(\mathfrak{D})$ is a rational number whose numerator and denominator are clearly polynomial-time computable. So for a given ε we run the algorithm ignoring the last

$$t = \lfloor \log(\frac{\varepsilon}{I(\mathfrak{D})}) \rfloor \text{ bits.}$$

Because $\mathrm{opt}_{\mathcal{Q}}(\mathfrak{D}) \leqslant n \cdot \max_i\{u(i) \cdot L(\mathfrak{D}, \chi_i)\}$ we get that $\mathrm{opt}_{\mathcal{Q}}(\mathfrak{D})$ is polynomially bounded by n, $1/\varepsilon$ and $\max u(x)$. But $u(x)$ can become arbitrary large.

Let $\delta > 0$ and let $\Delta \subseteq A$ be the set of all $a \in A$ such that $u(a) > \frac{1}{\delta}$. Because $u(a) \cdot L(\mathfrak{D}, \chi_a) \leqslant L(\mathfrak{D}, f^*)$ it is clear that $L(\mathfrak{D}, \chi_a) \leqslant \delta \cdot L(\mathfrak{D}, f^*)$ for all $a \in \Delta$.

Consider the problem $\mathcal{Q}_\delta$. It is the same problem as $\mathcal{Q}$ except that we delete all points in Δ. It is straightforward that $\mathcal{Q}_\delta$ can be δ-approximated in polynomial-time using algorithm `ExSol` ignoring the last

$$t = \lfloor \log(\frac{\delta}{I(\mathfrak{D}|_{A \setminus \Delta})}) \rfloor \text{ bits of the weights } L(\mathfrak{D}, \chi_a), a \in A \setminus \Delta.$$

Let f_δ^* be an optimal solution and f_δ' the δ-approximation of $\mathcal{Q}_\delta$.

The problem $\mathcal{Q}_r$ (the "rest"-problem) is defined as follows. We are looking for a function $f : \Delta \to \mathbb{N}$ such that $L(\mathfrak{D}, f)$ becomes maximal and $G(\mathfrak{D}, f_\delta' + f) \leqslant B(\mathfrak{D})$. Define

$$\begin{aligned}
v : \Delta \to \mathbb{N} \;\text{ such that }& G(\mathfrak{D}, f_\delta' + v(a) \cdot \chi_a) \leqslant B(\mathfrak{D}), \text{ and} \\
& G(\mathfrak{D}, f_\delta' + (v(a)+1) \cdot \chi_a) > B(\mathfrak{D}) \text{ for all } a \in \Delta.
\end{aligned}$$

Clearly v is polynomial-time computable. Let f_r^* be an optimal solution of $\mathcal{Q}_r$ and let $d \in \Delta$ such that

$$(v(a)+1) L(\mathfrak{D}, \chi_a) \leqslant (v(d)+1) L(\mathfrak{D}, \chi_d) \text{ for all } a \in \Delta.$$

Thus Theorem 14 implies that

$$\sum_{a \in \Delta} f_r^*(a) \cdot L(\mathfrak{D}, \chi_a) \leqslant (v(d)+1) L(\mathfrak{D}, \chi_d).$$

Define $f'_r = v(d) \cdot \chi_d$.

We now show that $f' = f'_\delta + f'_r$ is a 2δ-approximate solution of $\mathcal{Q}$. Clearly $L(\mathfrak{D}, f^*) \leqslant L(\mathfrak{D}, f^*_\delta) + L(\mathfrak{D}, f^*_r)$.

$$\begin{aligned}\frac{L(\mathfrak{D}, f^*) - L(\mathfrak{D}, f')}{L(\mathfrak{D}, f^*)} &\leqslant \frac{L(\mathfrak{D}, f^*_\delta) + L(\mathfrak{D}, f^*_r) - L(\mathfrak{D}, f'_\delta) - L(\mathfrak{D}, f'_r)}{L(\mathfrak{D}, f^*)} \\ &= \frac{L(\mathfrak{D}, f^*_\delta) - L(\mathfrak{D}, f'_\delta)}{L(\mathfrak{D}, f^*)} + \frac{L(\mathfrak{D}, f^*_r) - L(\mathfrak{D}, f'_r)}{L(\mathfrak{D}, f^*)} \\ &\leqslant \delta + \frac{L(\mathfrak{D}, \chi_d)}{L(\mathfrak{D}, f^*)} \leqslant \delta + \delta = 2\delta.\end{aligned}$$

Let $\delta = \varepsilon/2$. It is now clear that we get f' in time polynomial in n and $1/\varepsilon$.

The case where f is bounded by ψ is very similar. We use the function s instead of u to define $\mathcal{Q}_\delta$ and the "rest"-problem $\mathcal{Q}_r$. $\mathcal{Q}_\delta$ can be δ-approximated and $\mathcal{Q}_r$ can be approximated with $(2n+2)\max_{a \in \Delta} L(\mathfrak{D}, \chi_a)$.
Let f' be the resulting approximate solution and define $\delta = \varepsilon/(2n+3)$. Then f' is an ε-approximation and it is computable in time polynomial in n and $1/\varepsilon$. □

Example 17. The Max Bounded Knapsack problem is the same problem as Max Knapsack except that the number each object can be taken is bounded by a function s. If s is definable by a formula ψ then clearly Max Bounded Knapsack is in Max Φ and thus it has an FPTAS.

In the absence of more natural problems we give an example that shows the scope of Max Φ.

Example 18. An input instance consists of a set of n objects $A = \{a_0, \ldots, a_{n-1}\}$, each one with a profit $p(a_i)$ and a weight $w(a_i)$. Each object can be taken arbitrary often with the following constraints:

$$\begin{aligned} G(\mathfrak{D}, f) := \prod_{a \in A} w(a)^{f(a)} &\leqslant \prod_{a \in A} w(a)^2 =: B(\mathfrak{D}) \\ \psi = &\forall x (f(x) \leqslant p(x) + w^2(x)). \end{aligned}$$

The goal is to maximize the total profit $\sum_{a \in A} f(a)p(a)$.
It is easy to verify that this problem is in Max Φ and thus has an FPTAS.

4 The class Min Φ

Using the results of the previous section we can also get approximation results for minimization problems. We define the class Min Φ which is the dual of Max Φ.

Definition 19. The class Min Φ consists of all minimization problems $\mathcal{Q}$ whose input instances are arithmetical structures $\mathfrak{D} = (\mathfrak{A}, \mathfrak{N}, W)$ over σ such that

$$\mathrm{opt}_{\mathcal{Q}}(\mathfrak{D}) = \min_g \Big\{ L(\mathfrak{D}, g) : I(\mathfrak{D}, g) \geqslant B(\mathfrak{D}) \wedge (\mathfrak{D}, g) \models \psi \Big\},$$

where $g : A^m \to \mathbb{N}$, L and I are linear functionals and B and ψ satisfy the same conditions as in Definition 8.

Example 20. Consider a student who wants to get a Ph.D. At the beginning of his study he receives from his professor n open questions, say $q_0, \dots, q_{n-1}$, and for each question q_i he gets the time $t(q_i)$ the professor estimates to solve the question and he gets the importance $c(q_i)$ of the question to the community, e.g., problem P vs. NP has time 300 years (?) and importance 10^6. To receive the degree the thesis must contain a total importance B. The student wants to finish fast. How long will it take?

It is easy to see that this problem is in MIN Φ. The input instance $\mathfrak{D} = (A, \mathfrak{N}, W)$ is the arithmetical structure consisting of the set $A = \{q_0, \dots, q_{n-1}\}$ of cardinality n, the secondary part is $\mathfrak{N}$ and W contains the two weight functions $t : A \to \mathbb{N}$ and $c : A \to \mathbb{N}$.

$$\mathrm{opt}_{\mathcal{Q}}(\mathfrak{D}) = \min_{g: A \to \mathbb{N}} \Big\{ \sum_{x \in A} g(x)t(x) : \sum_{x \in A} g(x)c(x) \geqslant B \wedge \forall x (g(x) \leqslant 1) \Big\}.$$

Theorem 21. *Every problem in* MIN Φ *has an* FPTAS.

Proof. Let $\mathcal{Q} \in$ MIN Φ. We will reduce $\mathcal{Q}$ to a problem $\mathcal{Q}'$ in MAX Φ. If g is bounded, let u be the function equal to this bound and without loss of generality we can assume that $L(\mathfrak{D}, u(a) \cdot \chi_a) < L(\mathfrak{D}, g^*)$ for all $a \in A$, where g^* is an optimal solution. Otherwise define

$$v : A \to \mathbb{N} \text{ such that } I(\mathfrak{D}, v(a) \cdot \chi_a) \geqslant B(\mathfrak{D}), \text{ and}$$
$$I(\mathfrak{D}, (v(a) - 1) \cdot \chi_a) < B(\mathfrak{D}) \text{ for all } a \in A.$$

Clearly $\min_a L(\mathfrak{D}, v(a) \cdot \chi_a) \geqslant L(\mathfrak{D}, g^*)$. Therefore define the function

$$u : A \to \mathbb{N} \text{ such that } I(\mathfrak{D}, u(a) \cdot \chi_a) \leqslant \min_{b \in A} L(\mathfrak{D}, v(b) \cdot \chi_b), \text{ and}$$
$$I(\mathfrak{D}, (u(a) + 1) \cdot \chi_a) > \min_{b \in A} L(\mathfrak{D}, v(b) \cdot \chi_b) \text{ for all } a \in A.$$

Clearly $g^* \sqsubseteq u \sqsubseteq v$. We define the maximization problem $\mathcal{Q}'$:

$$\mathrm{opt}_{\mathcal{Q}'}(\mathfrak{D}) = \max_f \Big\{ L(\mathfrak{D}, f) : I(\mathfrak{D}, f) \leqslant I(\mathfrak{D}, u) - B(\mathfrak{D}) \wedge (\mathfrak{D}, f) \models \psi \Big\},$$

where $(\mathfrak{D}, f) \models \psi$ iff $f \sqsubseteq u$. Clearly $\mathcal{Q}'$ is in MAX Φ.

Obviously $\mathrm{opt}_{\mathcal{Q}}(\mathfrak{D}) + \mathrm{opt}_{\mathcal{Q}'}(\mathfrak{D}) = L(\mathfrak{D}, g^*) + L(\mathfrak{D}, f^*) = L(\mathfrak{D}, u)$ and $g^* + f^* = u$.

We know that $\mathcal{Q}'$ has an FPTAS. So let f' be an ε'-approximation for $\mathcal{Q}'$, i.e.,

$$\frac{L(\mathfrak{D}, f^*) - L(\mathfrak{D}, f')}{L(\mathfrak{D}, f^*)} \leqslant \varepsilon'.$$

Define $g' = u - f'$. Clearly $L(\mathfrak{D}, g') \geqslant L(\mathfrak{D}, g^*)$. Note that $L(\mathfrak{D}, u)/L(\mathfrak{D}, g^*) \leqslant p(n)$ where $p(n)$ is a polynomial. What is the relative error of g'?

$$\frac{L(\mathfrak{D}, g') - L(\mathfrak{D}, g^*)}{L(\mathfrak{D}, g')} = \frac{L(\mathfrak{D}, f^*) - L(\mathfrak{D}, f')}{L(\mathfrak{D}, f^*)} \cdot \frac{L(\mathfrak{D}, f^*)}{L(\mathfrak{D}, g')}$$
$$\leqslant \varepsilon' \cdot \frac{L(\mathfrak{D}, f^*)}{L(\mathfrak{D}, g^*)} \leqslant \varepsilon' \cdot p(n) =: \varepsilon.$$

It is clear that we get g' in time polynomial in n and $1/\varepsilon$. □

Example 22. 2-PROCESSOR SCHEDULING is in MIN Φ.
The goal is to distribute n jobs, each one of length $\ell(j)$, to 2 identical processors such that the total running time is minimal. Thus $A = \{a_0, \dots, a_{n-1}\}$ and

$$\mathrm{opt}_Q(\mathfrak{D}) = \min_{g: A \to \mathbb{N}} \Big\{ \sum_{x \in A} g(x)\ell(x) : \sum_{x \in A} g(x)\ell(x) \geqslant B(\mathfrak{D}) \wedge \forall x (g(x) \leqslant 1) \Big\},$$

where $B(\mathfrak{D}) = \left\lceil \frac{1}{2} \sum_{x \in A} \ell(x) \right\rceil$.

References

1. S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and intractability of approximation problems. In *Proc. 33rd IEEE Symp. on Foundations of Computer Science*, 1992.
2. A. K. Chandra, D. S. Hirschberg, and C. K. Wong. Approximate Algorithms for some Generalized Knapsack Problems. *Theoretical Computer Science*, 3:293–304, 1976.
3. P. Crescenzi and V. Kann. A compendium of NP optimization problems. Technical Report SI/RR-95/02, Dipartimento di Scienze dell'Informazione, Università di Roma "La Sapienza", 1995.
4. R. Fagin. Generalized first–order spectra and polynomial–time recognizable sets. In R. M. Karp, editor, *Complexity of Computation, SIAM-AMS Proceedings, Vol. 7*, pages 43–73, 1974.
5. E. Grädel and Y. Gurevich. Metafinite Model Theory. In D. Leivant, editor, *Logic and Computational Complexity, International Workshop LCC '94, Indianapolis, IN, USA*, volume 960 of *LNCS*, pages 313–366. Springer-Verlag, 1995.
6. O. H. Ibarra and C. E. Kim. Fast Approximation Algorithms for the Knapsack and Sum of Subset Problems. *Journal of the ACM*, 22:463–468, 1975.
7. S. Khanna, R. Motwani, M. Sudan, and U. Vazirani. On Syntactic versus Computational Views of Approximability. In *Proc. 35th IEEE Symp. on Foundations of Computer Science*, pages 819–830, 1994.
8. Ph. G. Kolaitis and M. N. Thakur. Logical definability of NP-optimization problems. *Information and Computation*, 115(2):321–353, 1994.
9. Ph. G. Kolaitis and M. N. Thakur. Approximation properties of NP minimization classes. *Journal of Computer and System Sciences*, 50(3):391–411, 1995.
10. A. Malmström. *Logic and Approximation.* PhD thesis, RWTH Aachen, 1997.
11. C. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity.* Prentice Hall, New Jersey, 1982.
12. C. Papadimitriou and M. Yannakakis. Optimization, approximization and complexity classes. *Journal of Computer and System Sciences*, 43:425–440, 1991.
13. M. Zimand. Weighted NP Optimization Problems: Logical Definability and Approximation Properties. In *Proc. 10th IEEE Symp. on Structure in Complexity Theory*, pages 12–28, 1995.