UNIVERSITE PARIS DAUPHINE

Laboratoire d'Analyse et Modélisation de Systèmes pour l'Aide à la Décision

CNRS UMR 7024

CAHIER DU LAMSADE

227

Septembre 2005

Approximability preserving reductions

Giorgio AUSIELLO, Vangelis Th. PASCHOS

CENTRE NATIONAL DE LA RECHERCHE SCIENTIFIQUE

Approximability preserving reductions*

Giorgio Ausiello[1] Vangelis Th. Paschos[2]

[1] Dipartimento di Informatica e Sistemistica
Università degli Studi di Roma "La Sapienza"
ausiello@dis.uniroma1.it

[2] LAMSADE, CNRS UMR 7024 and Université Paris-Dauphine
paschos@lamsade.dauphine.fr

September 15, 2005

Abstract

We present in this paper a "tour d'horizon" of the most important approximation-preserving reductions that have strongly influenced research about structure in approximability classes.

1 Introduction

The technique of transforming a problem into another in such a way that the solution of the latter entails, somehow, the solution of the former, is a classical mathematical technique that has found wide application in computer science since the seminal works of Cook [10] and Karp [19] who introduced particular kinds of transformations (called *reductions*) with the aim of studying the computational complexity of combinatorial decision problems. The interesting aspect of a reduction between two problems consists in its twofold application: on one side it allows to transfer positive results (resolution techniques) from one problem to the other and, on the other side, it may also be used for deriving negative (hardness) results. In fact, as a consequence of such seminal work, by making use of a specific kind of reduction, the polynomial-time *Karp-reducibility*, it has been possible to establish a complexity partial order among decision problems, which, for example, allows us to state that, modulo polynomial time transformations, the SATISFIABILITY problem is as hard as thousands of other combinatorial decision problems, even though the precise complexity level of all these problems is still unknown.

Strictly associated with the notion of reducibility is the notion of completeness. Problems that are complete in a complexity class via a given reducibility are, in a sense, the hardest problems of such class. Besides, given two complexity classes $\mathbf{C}$ and $\mathbf{C}' \subseteq \mathbf{C}$, if a problem Π is complete in $\mathbf{C}$ via reductions that belong (preserve membership) to $\mathbf{C}'$, in order to establish whether $\mathbf{C}' \subset \mathbf{C}$ it is "enough" to assess the actual complexity of Π (informally we say that Π is a candidate to separate $\mathbf{C}$ and $\mathbf{C}'$).

In this chapter we will show that an important role is played by reductions also in the field of approximation of hard combinatorial optimization problems. In this context the kind of reductions which will be applied are called *approximation preserving reductions*. Intuitively, in the most simple case, an approximation preserving reduction consists of two mappings f

*Preliminary version of Chapter 7 to appear in T. Gongalez (ed.), *Approximation algorithms and metaheuristics*, Taylor and Francis.

and g: f maps an instance x of problem Π into an instance $f(x)$ of problem Π', g maps back a feasible solution y of Π' into a feasible solution $g(y)$ of Π with the property that $g(y)$ is an approximate solution of problem Π whose quality is almost as good as the quality of the solution y for problem Π'. Clearly, again in this case, the role of an approximation preserving reduction is twofold: on one side it allows to transfer an approximation algorithm from problem Π' to problem Π; on the other side, if we know that problem Π cannot be approximated beyond a given threshold, such limitation applies also to problem Π'.

Various kinds of approximation preserving reducibilities will be introduced in this chapter and we will show how they can be exploited in a positive way, to transform solution heuristics from a problem to another and how, on the contrary, they may help in proving negative, inapproximability results.

It is well known that **NP**-hard combinatorial optimization problems behave in a very different way with respect to approximability and can be classified accordingly. While for some problems there exist polynomial-time approximation algorithms that provide solutions with a constant approximation ratio w.r.t. the optimum solution, for some other problems even a remotely approximate solution is computationally hard to achieve. Analogously to what happens in the case of the complexity of decision problems, approximation preserving reductions allow to establish a partial order among optimization problems in terms of approximability properties, independently from the actual level of approximation that for such problems can be achieved (and that in some cases is still undefined). Approximation preserving reductions can also be used to define complete problems which play an important role in the study of possible separations between approximation classes. The discovery that a problem is complete in a given approximation class provides a useful insight in understanding what makes a problem not only computationally hard but also resilient to approximate solutions.

As a final remark on the importance of approximation preserving reductions, let us observe that such reductions require that some correspondence between the combinatorial structure of two problems is established. This is not the case for reductions between decision problems. For example, in such case, we see that all **NP**-complete decision problems turn out to be mutually interreducible by means of polynomial-time reduction while when we consider the corresponding optimization problems, the different approximability properties, come to evidence. As a consequence, we can say that approximation preserving reductions are also a useful tool to analyze the deep relation existing between combinatorial structure of problems and the hardness of approximation.

The rest of this chapter is organized as follows. The next section is devoted to basic definitions and preliminary results concerning reductions among combinatorial optimization problems. In Section 3 we provide the first, simple example of approximation preserving reducibility, namely the *linear reducibility*, that while not as powerful as the reducibilities that will be presented in the sequel is widely used in practice. In Section 4 we introduce the reducibility that, historically, has been, the first to be introduced, *the strict reducibility* and we discuss the first completeness results based on reductions of such kind. Next, in Section 5 we introduce AP-reduction, and in Section 6 we discuss more extensive completeness results in approximation classes. In Section 7, we present a new reducibility, called FT-reducibility, that allows **PTAS**-completeness of natural **NPO** problems. Finally, in Section 8 we present other reductions with the specific aim of proving further inapproximability results. The last two sections of the chapter contain conclusions and references.

In this paper we assume that the reader is familiar with the basic notions of computational complexity regarding both decision problems and combinatorial optimization problems, as they are defined in [3, 25].

2 Basic definitions

Before introducing the first examples of reductions between optimization problems, let us recall the definitions of the basic notions of approximation theory and of the most important classes of optimization problems, characterized in terms of their approximability properties. First of all we introduce the class **NPO** which is the equivalent, for optimization problems, of the class of decision problems **NP**.

Definition 1. An ***NP*** *optimization problem,* ***NPO***, Π is defined as a four-tuple $(\mathcal{I}, \mathrm{Sol}, m, \mathrm{goal})$ such that:

- $\mathcal{I}$ is the set of *instances* of Π and it can be recognized in polynomial time;
- given $x \in \mathcal{I}$, $\mathrm{Sol}(x)$ denotes the set of *feasible solutions* of x; for any $y \in \mathrm{Sol}(x)$, $|y|$ (the size of y) is polynomial in $|x|$ (the size of x); given any x and any y polynomial in $|x|$, one can decide in polynomial time if $y \in \mathrm{Sol}(x)$;
- given $x \in \mathcal{I}$ and $y \in \mathrm{Sol}(x)$, $m(x, y)$ denotes the value of y and can be computed in polynomial time;
- $\mathrm{goal} \in \{min, max\}$ indicates the *type* of optimization problem. ∎

Given an **NPO** problem $\Pi = (\mathcal{I}, \mathrm{Sol}, m, \mathrm{goal})$ an optimum solution of an instance x of Π is usually denoted $y^*(x)$ and its measure $m(x, y^*(x))$ is denoted by $\mathrm{opt}(x)$.

Definition 2. Given an **NPO** problem $\Pi = (\mathcal{I}, \mathrm{Sol}, m, \mathrm{goal})$, an *approximation algorithm* $\mathtt{A}$ is an algorithm that given an instance x of Π returns a feasible solution $y \in \mathrm{Sol}(x)$. If $\mathtt{A}$ runs in polynomial time with respect to $|x|$, $\mathtt{A}$ is called a *polynomial-time approximation algorithm* for Π. ∎

The quality of the solution given by an approximation algorithm $\mathtt{A}$ for a given instance x is usually measured as the ratio $\rho_A(x)$, *approximation ratio*, between the value of the approximate solution, $m(x, A(x))$, and the value of the optimum solution $opt(x)$. For minimization problems, therefore, the approximation ratio is in $[1, \infty)$, while for maximization problems it is in $[0, 1]$.

Definition 3. An **NPO** problem Π belongs to the class **APX** if there exist a polynomial time approximation algorithm $\mathtt{A}$ and a value $r \in \mathbb{Q}$ such that, given any instance x of Π, $\rho_{\mathtt{A}}(x) \leqslant r$ (resp., $\rho_{\mathtt{A}}(x) \geqslant r$) if Π is a minimization problem (resp., a maximization problem). In such case $\mathtt{A}$ is called an r-approximation algorithm. ∎

Examples of combinatorial optimization problems belonging to the class **APX** are MAX SATISFIABILITY, MIN VERTEX COVER, MIN EUCLIDEAN TSP.

In some cases, a stronger form of approximability for **NPO** problems can be obtained by a *polynomial-time approximation scheme* (PTAS for short), that is a family of algorithms $\mathtt{A}_r$ such that, given any ratio $r \in \mathbb{Q}$, the algorithm $\mathtt{A}_r$ is an r-approximation algorithm whose running time is bounded by a suitable polynomial p as a function of $|x|$.

Definition 4. An **NPO** problem Π belongs to the class **PTAS** if there exists a polynomial-time approximation scheme $\mathtt{A}_r$ such that, given any $r \in \mathbb{Q}$, $r \neq 1$, and any instance x of Π, $\rho_{\mathtt{A}_r}(x) \leqslant r$ (resp., $\rho_{\mathtt{A}_r}(x) \geqslant r$) if Π is a minimization problem (resp., a maximization problem). ∎

Among the problems in **APX** listed above, the problem MIN EUCLIDEAN TSP can be approximated by means of a polynomial-time approximation scheme and hence belongs to the class

PTAS. Moreover, other examples of combinatorial optimization problems belonging to the class **PTAS** are MIN PARTITIONING and MAX INDEPENDENT SET ON PLANAR GRAPHS.

Finally, a stronger form of approximation scheme can be used for particular problems in **PTAS**, such as, for example, MAX KNAPSACK or MIN KNAPSACK. In such cases, in fact, the running time of the algorithm A_r is uniformly polynomial in r as made precise in the following definition.

Definition 5. An **NPO** problem Π belongs to the class **FPTAS** if there exists a polynomial-time approximation scheme $\mathtt{A}_r$ such that, given any $r \in \mathbb{Q}$, $r \neq 1$, and any instance x of Π, $\rho_{\mathtt{A}_r}(x) \leqslant r$ (resp., $\rho_{\mathtt{A}_r}(x) \geqslant r$) if Π is a minimization problem (resp., a maximization problem) and, furthermore, there exists a two variate polynomial q such that the running time of $\mathtt{A}_r(x)$ is bounded by $q(x, 1/(r-1))$ (resp., $q(x, 1/(1-r))$ in case of maximization problems). ∎

It is worth to remember that under the hypothesis that $\mathbf{P} \neq \mathbf{NP}$ all the above classes form a strict hierarchy that is $\mathbf{FPTAS} \subset \mathbf{PTAS} \subset \mathbf{APX} \subset \mathbf{NPO}$.

Let us note that there exist also other notorious approximability classes, as **Poly-APX**, **Log-APX**, **Exp-APX**, the classes of problems approximable within ratios that are, respectively, polynomials (or inverse of polynomials if goal = max) logarithms (or inverse of logarithms), exponentials (or inverse of exponentials) of the size of the input. The best studied among them is the class **Poly-APX**. Despite of their interest, for economy, these classes are not dealt in the chapter.

When the problem of characterizing approximation algorithms for hard optimization problems was tackled, soon the need arose for a suitable notion of reduction that could be applied to optimization problems in order to study their approximability properties ([17]).

> What is it that makes algorithms for different problems behave in the same way? Is there some stronger kind of reducibility than the simple polynomial reducibility that will explain these results, or are they due to some structural similarity between the problems as we define them?

Approximation preserving reductions provide an answer to the above question. Such reductions have an important role when we wish to assess the approximability properties of an **NPO** optimization problem and locate its position in the approximation hierarchy. In such case, in fact, if we can establish a relationship between the given problem and other known optimization problems, we can derive both positive information on the existence of approximation algorithms (or approximation schemes) for the new problem or, on the other side, negative information, showing intrinsic limitations to approximability. With respect to reductions between decision problems, reductions between optimization problems have to be more elaborate. Such reductions, in fact, have to map both instances and solutions of the two problems, and they have to preserve, so to say, the optimization structure of the two problems.

The first examples of reducibility among optimization problems were introduced by Ausiello, d'Atri and Protasi in [5, 6] and by Paz and Moran in [26]. In particular in [6] the notion of *structure preserving reducibility* is introduced and for the first time the completeness of MAX WSAT in the class of **NPO** problems is proved. Still it took a few more years until suitable notions of approximation preserving reducibilities were introduced by Orponen and Mannila in [22]. In particular their paper presented the strict reduction (see Section 4) and provided the first examples of natural problems who are complete under approximation preserving reductions: (MIN WSAT, MIN 0-1 LINEAR PROGRAMMING and MIN TSP).

Before introducing specific examples of approximation preserving reduction in the next sections, let us explain more formally how reductions between optimization problems can be defined,

starting from the notion of *basic reducibility* (called R-reducibility in the following, denoted $\leqslant_{\mathsf{R}}$) which underlays the most of the reducibilities that will be later introduced.

Definition 6. Let Π_1 and Π_2 be two **NPO** maximization problems. Then we say that $\Pi_1 \leqslant_{\mathsf{R}} \Pi_2$ if there exist two polynomial time computable functions f, g that satisfy the following properties:

- $f : \mathcal{I}_{\Pi_1} \rightarrow \mathcal{I}_{\Pi_2}$ such that $\forall x_1 \in \mathcal{I}_{\Pi_1}$, $f(x_1) \in \mathcal{I}_{\Pi_2}$; in other words, given an instance x_1 in Π_1, f allows to build an instance $x_2 = f(x_1)$ in Π_2;
- $g : \mathcal{I}_{\Pi_1} \times \mathrm{Sol}_{\Pi_2} \rightarrow \mathrm{Sol}_{\Pi_1}$ such that, $\forall (x_1, y_2) \in (\mathcal{I}_{\Pi_1} \times \mathrm{Sol}_{\Pi_2}(f(x_1)))$, $g(x_1, y_2) \in \mathrm{Sol}_{\Pi_1}(x_1)$; in other words, starting from a solution y_2 of the instance x_2, g determines a solution $y_1 = g(x_1, y_2)$ of the initial instance x_1. ∎

As we informally said in the introduction the aim of an approximation preserving reduction is to guarantee that if we achieve a certain degree of approximation in the solution of problem Π_2, then a suitable degree of approximation is reached for problem Π_1. As we will see, the various notions of approximation preserving reducibilities that will be introduced in the following, essentially differ in the mapping that is established between the approximation ratios of the two problems.

Before closing this section, let us introduce the notion of *closure* of a class of problems under a given type of reducibility. In what follows, given two **NPO** problems Π et Π' and a reducibility X, we will generally use the notation $\Pi \leqslant_{\mathsf{X}} \Pi'$ to indicate that Π reduces to Π' via reduction of type X.

Definition 7. Let $\mathbf{C}$ be a class of **NPO** problems and X a reducibility. Then, the closure $\overline{\mathbf{C}}^{\mathsf{X}}$ of $\mathbf{C}$ under X is defined as: $\overline{\mathbf{C}}^{\mathsf{X}} = \{\Pi \in \mathbf{NPO} : \exists \Pi' \in \mathbf{C}, \Pi \leqslant_{\mathsf{X}} \Pi'\}$. ∎

3 The linear reducibility

The first kind of approximation preserving reducibility that we want to show is a very natural and simple transformation among problems which consists in two linear mappings, one between the values of the optimum solutions of the two problems and one between the errors of the corresponding approximate solutions: the *linear reducibility* (L-reducibility, denoted $\leqslant_{\mathsf{L}}$).

Definition 8. Let Π and Π' be two problems in **NPO**. Then, we say that $\Pi_1 \leqslant_{\mathsf{L}} \Pi_2$, if there exist two functions f and g (basic reduction) and two constants $\alpha_1 > 0$ and $\alpha_2 > 0$ such that $\forall x \in \mathcal{I}_{\Pi}$ and $\forall y' \in \mathrm{Sol}_{\Pi'}(f(x))$:

- $\mathrm{opt}_{\Pi'}(f(x)) \leqslant \alpha_1 \mathrm{opt}_{\Pi}(x)$;
- $|m_{\Pi}(x, g(y')) - \mathrm{opt}_{\Pi}(x)| \leqslant \alpha_2 |m_{\Pi'}(f(x), y') - \mathrm{opt}_{\Pi'}(f(x))|$. ∎

This type of reducibility has been introduced in [24] and has played an important role in the characterization of the hardness of approximation. In fact it is easy to observe that the following property holds.

Fact 1. Given two problems Π and Π', if $\Pi \leqslant_{\mathsf{L}} \Pi'$ and $\Pi' \in \mathbf{PTAS}$, then $\Pi \in \mathbf{PTAS}$. In other words, the L-reduction preserves membership in **PTAS**.

Example 1. MAX 3-SAT $\leqslant_{\mathsf{L}}$ MAX 2-SAT. Let us consider an instance φ with m clauses (w.l.o.g., let us assume that all clauses consist of exactly three literals); let l_i^1, l_i^2 and l_i^3, be the three literals of of the i-th clause, $i = 1, \ldots, m$. To any clause we associate the ten following new clauses, each one consisting of at most two literals: l_i^1, l_i^2, l_i^3, l_i^4, $\bar{l}_i^1 \vee \bar{l}_i^2$, $\bar{l}_i^1 \vee \bar{l}_i^3$, $\bar{l}_i^2 \vee \bar{l}_i^3$, $l_i^1 \vee \bar{l}_i^4$, $l_i^2 \vee \bar{l}_i^4$, $l_i^3 \vee \bar{l}_i^4$, where l_i^4 is a new variable. Let C_i' be the conjunction of the ten clauses derived

from clause C_i. The formula $\varphi' = f(\varphi)$ is the conjunction of all clauses C'_i, $i = 1, \ldots, m$, i.e., $\varphi' = f(\varphi) = \wedge_{i=1}^{m} C'_i$ and it is an instance of MAX 2-SAT.

It is easy to see that all truth assignments for φ' satisfy at most seven clauses in any C'_i. On the other side, for any truth assignment for φ satisfying C_i, the following truth assignment for l_i^4 is such that the extended truth assignment satisfies exactly seven clauses in C'_i: if exactly one (resp., all) of the variables l_i^1, l_i^2, l_i^3 is (resp., are) set to **true**, then l_i^4 is set to **false** (resp., **true**); otherwise (exactly one literal in C_i is set to **false**), l_i^4 can be indifferently **true** or **false**. Finally, if C_i is not satisfied (l_i^1, l_i^2 and l_i^3 are all set to **false**), no truth assignment for l_i^4 can satisfy more than six clauses of C'_i while six are guaranteed by setting l_i^4 to **false**. This implies that $\mathrm{opt}(\varphi') = 6m + \mathrm{opt}(\varphi) \leqslant 13\mathrm{opt}(\varphi)$ (since $m \leqslant 2\mathrm{opt}(\varphi)$, see Lemma 2 in Section 6.2).

Given a truth assignment for φ', we consider its restriction $\tau = g(\varphi, \tau')$ on the variables of φ; for such assignment τ we have: $m(\varphi, \tau) \geqslant m(\varphi', \tau') - 6m$. Then:

$$\mathrm{opt}(\varphi) - m(\varphi, \tau) = \mathrm{opt}\left(\varphi'\right) - 6m - m(\varphi, \tau) \leqslant \mathrm{opt}\left(\varphi'\right) - m\left(\varphi', \tau'\right)$$

This means that the reduction we have defined is an L-reduction with $\alpha_1 = 13$ and $\alpha_2 = 1$. ∎

L-reductions provide a simple way to prove hardness of approximability. An immediate consequence of the reduction that has been shown above and of Fact 1 is that, since MAX 3-SAT does not allow a PTAS (see [3, 25]) so does MAX 2-SAT. The same technique can be used to show the non existence of PTAS for a large class of optimization problems, among others MAX CUT, MAX INDEPENDENT SET-B (that is, MAX INDEPENDENT SET on graphs with bounded degree), MIN VERTEX COVER etc.

Before closing this section, let us observe that the set of ten 2-SAT clauses that we have used in Example 1 for constructing the 2-SAT formula $\varphi' = f(\varphi)$, is strongly related to the bound on approximability established in the example. Really, the proof of the result is based on the fact that at least six out of the ten clauses can always be satisfied while exactly seven out of ten can be satisfied, if and only if the original 3-SAT clause is satisfied. A combinatorial structure of this kind, which allows to transfer (in)approximability results from a problem to another, is called a *gadget* (see [29]). The role of gadgets in approximation preserving reductions will be discussed further in Section 8.

4 Strict reducibility and complete problems in NPO

As we informally said in the introduction, an important characteristic of an approximation preserving reduction from a problem Π_1 to a problem Π_2 is that the solution y_1 of problem Π_1 produced by the mapping g should be at least as good as the original solution y_2 of problem Π_2. This property is not necessarily true for any approximation preserving reduction (it is easy to observe that, for example L-reductions do not always satisfy it) but it is true for the most natural reductions that have been introduced in the early phase of approximation studies: the *strict reductions* [22].

In the following, we present the strict reducibility (S-reducibility, denoted $\leqslant_{\mathsf{S}}$) referring to minimization problems but the definition can be trivially extended to all types of optimization problems.

Definition 9. Let Π_1 and Π_2 be two **NPO** minimization problems. Then, we say that $\Pi_1 \leqslant_{\mathsf{S}} \Pi_2$ if there exist two polynomial time computable functions f, g that satisfy the following properties:

- f and g are defined as in a basic reduction;
- $\forall x \in \mathcal{I}_{\Pi_1}$, $\forall y \in \mathrm{Sol}_{\Pi_2}(f(x))$, $\rho_{\Pi_2}(f(x), y) \geqslant \rho_{\Pi_1}(x, g(x, y))$. ∎

It is easy to observe that the S-reducibility preserves both membership in **APX** and in **PTAS**.

Property 1. Given two minimization problems Π_1 and Π_2, if $\Pi_1 \leqslant_{\mathsf{S}} \Pi_2$ and $\Pi_2 \in$ **APX** (resp., $\Pi_2 \in$ **PTAS**), then $\Pi_1 \in$ **APX** (resp., $\Pi_2 \in$ **PTAS**). ∎

Example 2. Consider the MIN WEIGHTED VERTEX COVER problem in which the weights of vertices are bounded by a polynomial $p(n)$ and let us prove that this problem S-reduces to the unweighted MIN VERTEX COVER problem. Let us consider an instance $(G(V,E),\vec{w})$ of the former and let us see how it can be transformed into an instance $G'(V',E')$ of the latter. We proceed as follows: for any vertex $v_i \in V$, with weight w_i, we construct an independent set W_i of w_i new vertices in V'; next, for any edge $(v_i,v_j) \in E$, we construct a complete bipartite graph among the vertices of the independent sets W_i et W_j in G'. This transformation is clearly polynomial since the resulting graph G' has $\sum_{i=1}^{n} w_i \leqslant np(n)$ vertices.

Let us now consider a cover C' of G' and, w.l.o.g., let us assume it is minimal w.r.t. inclusion (in case it is not, we can easily delete vertices until we reach a minimal cover). We claim that at this point C' has the form: $\cup_{j=1}^{\ell} W_{i_j}$, i.e., there is an ℓ such that C' consists of ℓ independent sets W_i. Suppose that the claim is not true. Let us consider an independent set W_k which is only partially included in C' (that is a non empty portion W'_k of it belongs to C'). Let us also consider all independent sets W_p that are entirely or partially included in C' and moreover are connected by edges to the vertices of W_k. Two cases may arise: (i) all considered sets W_p have their vertices included in C'; in this case the existence of W'_k would contradict the minimality of C'; (ii) among the considered sets W_p there is at least one set W_q out of which only a non empty portion W'_q is included in C'; in this case, since the subgraph of G' induced by $W_k \cup W_q$ is a complete bipartite graph, the edges connecting the vertices of $W_p \setminus W'_p$ with the vertices of $W_q \setminus W'_q$ are not covered by C' and this would contradict the assumption that C' is a cover of G'. As a consequence, the size of C' satisfies $|C'| = \sum_{j=1}^{\ell} w_{i_j}$ and the function g of the reduction can then be defined as follows: *if C' is a cover of G' and if W_i, $i = 1,\ldots,\ell$, are the independent sets that form C', then a cover C for G contains all corresponding vertices $v_1,\ldots,v_\ell$ of V*. Clearly g can be computed in polynomial time.

From these premises we can immediately infer that the same approximation ratio that is guaranteed for A on G' is also guaranteed by g on G. The shown reduction is hence an S-reduction. ∎

An immediate corollary of the strict reduction shown in the example is that the approximation ratio 2 for MIN VERTEX COVER (that we know can be achieved by various approximation techniques, see [16]) also holds for the weighted version of the problem, dealt in Example 2.

The S-reducibility is indeed a very strong type of reducibility: in fact it requires a strong similarity between two optimization problems and it is not easy to find problems that exhibit such similarity. The interest for the S-reducibility arises mainly from the fact that by making use of reductions of this kind, Orponen and Mannila have identified the first optimization problem that is complete in the class of **NPO** minimization problems: the problem MIN WSAT. Let us consider a Boolean formula in conjunctive normal form φ over n variables $x_1,\ldots,x_n$ and m clauses. Any variable x_i has a positive weight $w_i = w(x_i)$. Let us assume that the truth assignment that puts all variables to **true** is feasible, even if it does not satisfy φ. Besides, let us assume that t_i is equal to 1 if τ assigns value **true** to the i-th variable and 0 otherwise. We want to determine the truth assignment τ of φ which minimizes: $\sum_{i=1}^{n} w_i t_i$. The problem MAX WSAT can be defined in similar terms. In this case we assume that the truth assignment that puts all variables to **false** is feasible and we want to determine the truth assignment τ that maximizes: $\sum_{i=1}^{n} w_i t_i$. In the variants MIN W3-SAT and MAX W3-SAT, we consider that all clauses contain exactly 3 literals.

The fact that MIN WSAT is complete in the class of **NPO** minimization problems under S-reductions implies that this problem does not allow any constant ratio approximation (unless $\mathbf{P} = \mathbf{NP}$) ([6, 26, 22]). In fact, due to the properties of S-reductions, if a problem which is complete in the class of **NPO** minimization problems problem was approximable then all **NPO** minimization problems would. Since it is already known that some minimization problems in **NPO** do not allow any constant ratio approximation algorithm (namely MIN TSP on general graphs), then we can deduce that (unless $\mathbf{P} = \mathbf{NP}$) no complete problem in the class of **NPO** minimization problems allows any constant ratio approximation algorithm.

Theorem 1. MIN WSAT *is complete in the class of minimization problems belonging to* ***NPO*** *under S-reductions.*

Sketch of proof. The proof is based on a modification of Cook's proof of the **NP**-completeness of SAT [10]. Let us consider a minimization problem $\Pi \in \mathbf{NPO}$, the polynomial p which provides the bounds relative to problem Π (see Definition 1) and an instance x of Π. The following non deterministic Turing machine M (with two output tapes T_1 and T_2) generates all feasible solutions $y \in \mathrm{Sol}(x)$ together with their values:

- generate y, such that $|y| \leqslant p(|x|)$;
- if $y \notin \mathrm{Sol}(x)$, then reject; otherwise, write y on output tape T_1, $m(x, y)$ on output tape T_2 and accept.

Let us now consider the reduction that is currently used in the proof of Cook's theorem (see [23]) and remember that such reduction produces a propositional formula in conjunctive normal form that is satisfied if and only if the computation of the Turing machine accepts. Let φ_x be such formula and $x_n, x_{n-1}, \ldots, x_0$ the variables of φ_x that correspond to the cells of tape T_2 where M writes the value $m(x, y)$ in binary (w.l.o.g., we can assume such cells to be consecutive), such that a satisfying assignment of φ_x, x_i is **true** if and only if the $(n-i)$-th bit of $m(x, y)$ is equal to 1. Given an instance x of Π the function f of the S-reduction provides an instance of MIN WSAT consisting of the pair (φ_x, ψ) where $\psi(x) = \psi(x_i) = 2^i$, for $i = 0, \ldots, n$ and $\psi(x) = 0$, for any other variable x in φ_x.

The function g of the S-reduction is defined as follows. For any instance x of Π and any solution $\tau' \in \mathrm{Sol}(f(x))$ (i.e., any truth assignment τ' which satisfies the formula φ_x (for simplicity we only consider the case in which the formula φ_x is satisfiable)), we recover from φ_x the representation of the solution y written on tape T_1. Besides, we have that:

$$m\left(x, g\left(x, \tau'\right)\right) = \sum_{\tau'(x_i)=\mathbf{true}} 2^i = m\left((\varphi_x, \psi), \tau'\right)$$

where by $\tau'(x_i)$ we indicate the value of variable x_i according to the assignment τ'. As a consequence we have: $r(x, g(x, \tau')) = r(f(x), \tau')$ and the described reduction is an S-reduction. ∎

After having established that MIN WSAT is complete for **NPO** minimization problems under the S-reducibility we can then proceed to find other complete problems in this class.

Let us consider the following definition of the MIN 0-1 LINEAR PROGRAMMING problem (the problem MAX 0-1 LINEAR PROGRAMMING can be defined analogously). We consider a matrix $A \in \mathbb{Z}^{m \times n}$ and two vectors $\vec{b} \in \mathbb{Z}^m$ and $\vec{w} \in \mathbb{N}^n$. We want to determine a vector $\vec{y} \in \{0, 1\}^n$ that verifies $A\vec{y} \geqslant \vec{b}$ and minimizes the quantity $\vec{w} \cdot \vec{y}$.

Clearly MIN 0-1 LINEAR PROGRAMMING is an **NPO** minimization problem. The reduction from MIN WSAT to MIN 0-1 LINEAR PROGRAMMING is a simple modification of the standard reduction among the corresponding decision problems. Suppose that the following instance of MIN 0-1 LINEAR PROGRAMMING, consisting of a matrix $A \in \mathbb{Z}^{m \times n}$ and two vectors $\vec{b} \in \mathbb{Z}^m$

and $\vec{w} \in \mathbb{N}^n$, is the image $f(x)$ of an instance x of MIN WSAT and suppose that $\vec{y}$ is a feasible solution of $f(x)$ whose value is $m(f(x), \vec{y}) = \vec{w} \cdot \vec{y}$. Then, $g(x, \vec{y})$ is a feasible solution of x, that is a truth assignment τ, whose value is $m(x, \tau) = \sum_{i=1}^{n} w_i t_i$ where t_i is equal to 1 if τ assigns value **true** to the i-th variable and 0 otherwise. Since we have $\sum_{i=1}^{n} w_i t_i = \vec{w} \cdot \vec{y}$ it is easy to see that the reduction (f, g, c), where c is the identity function, is an S-reduction[1] and, as a consequence, MIN 0-1 LINEAR PROGRAMMING is also complete in the class of **NPO** minimization problems.

It is not difficult to prove that an analogous result holds for maximization problems, that is, MAX WSAT is complete under S-reductions in the class of **NPO** maximization problems.

At this point of the chapter we still do not have the technical instruments to establish a more powerful result, that is, to identify problems which are complete under S-reductions for the entire class of **NPO** problems. In order to prove such result we need to introduce a more involved kind of reducibility, the AP-reducibility (see Section 5). In fact, by means of AP-reductions MAX WSAT can itself be reduced to MIN WSAT and viceversa (see [12]) and therefore it can be shown that (under AP-reductions) both problems are indeed **NPO**-complete.

5 AP-reducibility

After the seminal paper by Orponen and Mannila [22] research on approximation preserving reducibility was further developed (see, for example, [27, 28, 21]); nevertheless, the beginning of the structural theory of approximability of optimization problems can be traced back to the fundamental paper by Crescenzi and Panconesi [13] where reducibilities preserving membership in **APX** (A-reducibility), **PTAS** (P-reducibility) and **FPTAS** (F-reducibility) were studied and complete problems for each of the three kinds of reducibilities were shown, respectively in **NPO**, **APX** and **PTAS**. Unfortunately the problems which are proved complete in **APX** and in **PTAS** in this paper are quite artificial.

Along a different line of research, during the same years, the study of logical properties of optimization problems has led Papadimitriou and Yannakakis ([24]) to the syntactic characterization of an important class of approximable problems, the class **Max-SNP**. Completeness in **Max-SNP** has been defined in terms of L-reductions (see Section 3) and natural complete problems (e.g., MAX 3-SAT, MAX 2-SAT, MIN VERTEX COVER-B etc.) have been found. The relevance of such approach is related to the fact that it is possible to prove that **Max-SNP**-complete problems do not allow PTAS (unless $\mathbf{P} = \mathbf{NP}$).

The two approaches have been reconciled by Khanna, Motwani, Sudan and Vazirani, in [20], where the closure of syntactically defined classes with respect to an approximation preserving reduction were proved equal to the more familiar computationally defined classes. As a consequence of this result any **Max-SNP**-completeness result appeared in the literature can be interpreted as an **APX**-completeness result. In this paper a new type of reducibility is introduced, the E-reducibility. With respect to the L-reducibility, in the E-reducibility the constant α_1 is replaced by a polynomial $p(|x|)$. This reducibility is fairly powerful since it allows to prove that MAX 3-SAT is complet for **APX-PB** (the class of problems in **APX** whose values are bounded by a polynomial in the size of the instance) such as MAX 3-SAT. On the other side it remains somewhat restricted because it does not allow the transformation of **PTAS** problems (such as MAX KNAPSACK) into problems belonging to **APX-PB**.

The final answer to the problem of finding the suitable kind of reducibility (powerful enough to establish completeness results both in **NPO** and **APX**) is the AP-reducibility introduced by Crescenzi, Kann, Silvestri and Trevisan in [12].

In fact, the types of reducibility that we have introduced so far (linear and strict reducibilities) suffer from various limitations. In particular we have seen that strict reductions allow us to prove

[1] Note that, in this case, the reduction is also a linear reductions with $\alpha = \beta = 1$

the completeness of MIN WSAT in the class of **NPO** minimization problems but are not powerful enough to allow the identification of problems which are complete for the entire class **NPO**. Besides, both linear and strict reductions, in different ways, impose strong constraints on the values of the solutions of the problems among which the reduction is established.

In this section, we provide the definition of the AP-reducibility (denoted $\leqslant_{\mathsf{AP}}$) and we illustrate its properties. Completeness results in **NPO** and in **APX** based on AP-reductions are shown in Section 6.

Definition 10. Let Π_1 and Π_2 be two minimization **NPO** problems. An AP-reduction between Π_1 et Π_2 is a triple (f, g, α), where f and g are functions and α is a constant, such that, for any $x \in \mathcal{I}_{\Pi_1}$ and $r > 1$:

- $f(x, r) \in \mathcal{I}_{\Pi_2}$ is computable in time $t_f(|x|, r)$ polynomial in $|x|$ for a fixed r; $t_f(n, \cdot)$ is non increasing;
- for any $y \in \mathrm{Sol}_{\Pi_2}(f(x, r))$, $g(x, y, r) \in \mathrm{Sol}_{\Pi_1}(x)$ is computable in time $t_g(|x|, y, r)$ which is polynomial both in $|x|$ and in $|y|$ for an fixed r; $t_g(n, n, \cdot)$ is non increasing;
- for any $y \in \mathrm{Sol}_{\Pi_2}(f(x, r))$, $\rho_{\Pi_2}(f(x, r), y) \leqslant r$ implies $\rho_{\Pi_1}(x, g(x, y, r)) \leqslant 1 + \alpha(r - 1)$. ∎

It is worth to underline the main differences of AP-reductions with respect to the reductions introduced until now. In first place, with respect to L-reductions the constraint that the optimum values of the two problems are linearly related has been dropped. In second place, with respect to the S-reductions we allow a weaker relationship to hold between the approximation ratios achieved for the two problems. Besides, an important condition which is needed in the proof of **APX**-completeness is that, in AP-reductions, the two functions f and g may depend on the approximation ratio r. Such extension is somewhat natural since there is no reason to ignore the quality of the solution we are looking for, when reducing one optimization problem to another and it plays a crucial role in the completeness proofs. On the other side, since in many applications such knowledge is not required, whenever functions f and g do not use the dependency on r, we will avoid specifying this dependency. In other words, we will write $f(x)$ and $g(x, y)$ instead of $f(x, r)$ and $g(x, y, r)$, respectively.

Proposition 1. *Given two minimization problems* Π_1 *and* Π_2*, if* $\Pi_1 \leqslant_{\mathsf{AP}} \Pi_2$ *and* $\Pi_2 \in$ ***APX*** *(resp.,* $\Pi_2 \in$ ***PTAS****), then* $\Pi_1 \in$ ***APX*** *(resp.,* $\Pi_1 \in$ ***PTAS****).*

Sketch of proof. Let (f, g, α) be such reduction. Assume $\Pi_2 \in$ **APX**; let $\mathtt{A}_2$ be an approximation algorithm for Π_2 which guarantees an approximation ratio smaller than, or equal to, r_2. Then, by definition 10, the algorithm $\mathtt{A}_1(x) = g(x, \mathtt{A}_2(f(x, r)), r)$ is an approximation algorithm for Π_1 which guarantees an approximation ratio $r_1 \leqslant 1 + \alpha(r_2 - 1)$. This is a constant, not depending on x, since so are r_2 and α.

Analogously, if $\Pi_2 \in$ **PTAS** and $\mathtt{A}_2$ is a polynomial time approximation scheme for Π_2, then $\mathtt{A}_1(x, r_2) = g(x, \mathtt{A}_2(f(x, r_1), r_1), r_1)$ is a polynomial time approximation scheme for Π_1 with $r_1 = 1 + ((r_2 - 1)/\alpha)$. ∎

As a last remark, let us observe that the S-reducibility is a particular case of AP-reducibility, corresponding to the case in which $\alpha = 1$. More generally, the AP-reducibility is sufficiently broad to encompass almost all known approximation preserving reducibilities while maintaining the property of establishing a linear relation between performance ratios: this is important in order to preserve membership in all approximation classes.

6 NPO-completeness and APX-completeness

6.1 NPO-completeness

In the preceding section, we have announced that by means of a suitable type of reduction we can transform an instance of MAX WSAT into an instance of MIN WSAT. This can now be obtained by making use of AP-reductions. By combining this result with Theorem 1 and with the corresponding result concerning the completeness of MAX WSAT in the class of **NPO**-maximization problems, we can assemble the complete proof that MIN WSAT is complete for the entire class **NPO** under AP-reductions. The inverse reduction, from MIN WSAT to MAX WSAT can be shown in a similar way, leading to the proof that also MAX WSAT is complete for the entire class **NPO** under AP-reductions.

Theorem 2. MAX WSAT *can be AP-reduced to* MIN WSAT *and vice-versa.*

Sketch of proof. The proof works as follows. First a simple reduction can be defined which transforms a given instance φ of MAX WSAT into an instance φ' of MIN WSAT with α depending on r. Such reduction can then be modified into a real AP-reduction in which α is a constant, not depending on r, while, of course, the functions f and g will depend on r. We limit ourselves to describing the first step. The complete proof can be found in [3].

Let φ be the formula produced in the reduction proving the completeness of MAX WSAT for the class of **NPO** maximization problems. Then, $f(\varphi)$ be the formula $\varphi \wedge \alpha_1 \wedge \cdots \wedge \alpha_s$ where α_i is $z_i \equiv (\overline{v}_1 \wedge \cdots \wedge \overline{v}_{i-1} \wedge v_i)$, $z_1, \ldots, z_s$ are new variables with weights $w(z_i) = 2^i$ for $i = 1, \ldots, s$, and all other variables (even the v-variables) have zero weight. If τ is a satisfying truth assignment for $f(\varphi)$, let $g(\varphi, \tau)$ be the restriction of τ to the variables that occur in φ. This assignment clearly satisfies φ. Note that exactly one among the z-variables is **true** in any satisfying truth assignment of $f(\varphi)$. If all z-variables were **false**, then all v-variables would be **false**, which is not allowed. On the other hand, it is clearly not possible that two z-variables are **true**. Hence, for any feasible solution τ of $f(\varphi)$, we have that $m(f(\varphi), \tau) = 2^i$, for some i with $1 \leqslant i \leqslant s$. This finally implies: $2^{s-i} \leqslant m(\varphi, g(\varphi, \tau)) < 2 \cdot 2^{s-i}$ and since:

$$\begin{aligned} m(f(\varphi), \tau) = 2^i \Leftrightarrow z_i = 1 &\iff v_1 = v_2 = \cdots = v_{i-1} = 0, v_i = 1 \\ &\iff 2^{s-i} \leqslant m(\varphi, g(\varphi, \tau)) < 2 \cdot 2^{s-i} \end{aligned}$$

we have that, for any feasible solution τ of $f(\varphi)$:

$$\frac{2^s}{m(f(\varphi), \tau)} \leqslant m(\varphi, g(\varphi, \tau)) < 2\frac{2^s}{m(f(\varphi), \tau)}$$

This is in particular true for the optimal solution (observe that any satisfying truth assignment for φ can be easily extended to a satisfying truth assignment for $f(\tau)$). Thus, after some easy algebra, the performance ratio of $g(\varphi, \tau)$ with respect to φ verifies: $r(\varphi, g(\varphi, \tau)) > 1/(2r(f(\varphi), \tau))$.

Thus, the performance ratio of $g(\varphi, \tau)$ with respect to φ is:

$$\begin{aligned} r(\varphi, g(\varphi, \tau)) &= \frac{m(\varphi, g(\varphi, \tau))}{\mathrm{opt}(\varphi)} > \frac{\frac{2^s}{m(f(\varphi), \tau)}}{2\frac{2^s}{\mathrm{opt}(f(\varphi))}} \\ &= \frac{1}{2}\frac{\mathrm{opt}(f(\varphi))}{m(f(\varphi), \tau)} = \frac{1}{2}\frac{1}{r(f(\varphi), \tau)} \end{aligned}$$

The reduction satisfies the approximation preserving condition with a factor $\alpha = (2r-1)/(r-1)$. In order to obtain a factor α not depending on r the reduction can be modified by introducing 2^k more variables for a suitable integer k. ∎

Other problems that have been shown **NPO**-complete are MIN (MAX) W3-SAT and MIN TSP ([22]). As it has been observed before, as a consequence of their **NPO**-completeness under approximation preserving reductions, for all these problems does not exist any r-approximate algorithm with constant r, unless $\mathbf{P} = \mathbf{NP}$.

6.2 APX-completeness

As it has been mentioned above, the existence of an **APX**-complete problem has already been shown in [13] (see also [4]) but the problem that is proved complete in such framework is a rather artificial version of MAX WSAT. The reduction used in such result is called P-reduction. Unfortunately no natural problem has been proved complete in **APX** using the same approach. In this section, we prove the **APX**-completeness under AP-reduction of a natural and popular problem: MAX 3-SAT. The proof is crucially based on the following two lemmas (whose proofs are not provided in this paper).

The first lemma is proved in [1] and is based on a powerful algebraic technique for the representation of propositional formulæ (see also [3]), while the second one states a well known property of propositional formulæ and is proved in [3, 17].

Lemma 1. *There is a constant $\epsilon > 0$ and two functions f_s and g_s such that, given any propositional formula φ in conjunctive normal form, the formula $\psi = f_s(\varphi)$ is a conjunctive normal form formula with at most three literals per clause which satisfies the following property: for any truth assignment T' satisfying at least a portion $1 - \epsilon$ of the maximum number of satisfiable clauses in ψ, $g_s(\varphi, T')$ satisfies φ if and only if φ is satisfiable.*

Lemma 2. *Given a propositional formula in conjunctive normal form, at least one half of its clauses can always be satisfied.*

Theorem 3. MAX 3-SAT *is **APX**-complete.*

Sketch of proof. As it has been done in the case of the proofs of **NPO**-completeness, we split the proof in two parts. First, we show that MAX 3-SAT is complete in the class of **APX** maximization problems and then we show that any **APX** minimization problem can be reduced to an **APX** maximization problem. In order to make the proof easier, we adopt the convention used in [3]. The approximation ratio of a maximization problem in this context will be defined as the ratio between the value of the optimum solution $\mathrm{opt}(x)$ and the value of the approximate solution $m(x, \mathtt{A}(x))$. For both maximization and minimization problems, therefore, the approximation ratio is in $[1, \infty)$. Let us first observe that MAX 3-SAT $\in$ **APX** since it can be approximated up to the ratio 0.8006 ([29]).

Now we can sketch the proof that MAX 3-SAT is hard for the class of maximization problems in **APX**. Let us consider a maximization problem $\Pi \in$ **APX**. Let $\mathtt{A}_\Pi$ be a polynomial time r_Π-approximation algorithm for Π. In order to construct an AP-reduction, let us define the parameter α as follows: $\alpha = 2(r_\Pi \log r_\Pi + r_\Pi - 1) \times ((1+\epsilon)/\epsilon)$ where ϵ is the constant of Lemma1. Let us now chose $r > 1$ and let us consider the following two cases: $1 + \alpha(r-1) \geqslant r_\Pi$ and $1 + \alpha(r-1) < r_\Pi$.

In the case $1 + \alpha(r-1) \geqslant r_\Pi$, given any instance x of Π and given any truth assignment τ for MAX 3-SAT, we trivially define: $f(x, r)$ to be the empty formula and $g(x, \tau, r) = \mathtt{A}_\Pi(x)$. It can easily be seen that $r(x, g(x, \tau, r)) \leqslant r_\Pi \leqslant 1 + \alpha(r-1)$ and the reduction is an AP-reduction.

Let us then consider the case $1 + \alpha(r-1) < r_\Pi$ and let us define $r_n = 1 + \alpha(r-1)$; then:

$$r = \frac{r_n - 1}{\alpha} + 1$$

If we define $k = \lceil \log_{r_n} r_\Pi \rceil$, we can partition the interval $[m(x, \mathtt{A}_\Pi(x)), r_\Pi m(x, \mathtt{A}_\Pi(x))]$ in the following k subintervals:

$$\begin{aligned}
&[m(x, \mathtt{A}_\Pi(x)), r_n m(x, \mathtt{A}_\Pi(x))] \\
&\left[r_n^i m(x, \mathtt{A}_\Pi(x)), r_n^{i+1} m(x, \mathtt{A}_\Pi(x))\right], \quad i = 1, \ldots, k-2 \\
&\left[r_n^{k-1} m(x, \mathtt{A}_\Pi(x)), r_\Pi m(x, \mathtt{A}_\Pi(x))\right]
\end{aligned}$$

Then we have $m(x, \mathtt{A}_\Pi(x)) \leqslant \mathrm{opt}(x) \leqslant r_\Pi m(x, \mathtt{A}_\Pi(x)) \leqslant r_n^k m(x, \mathtt{A}_\Pi(x))$, i.e., the optimum value of instance x of Π belongs to one of the subintervals.

Note that by, definition, $k < (r_\Pi \log r_\Pi + r_\Pi - 1)/(r_n - 1)$ and by making use of the definitions of α, r and k, we obtain: $r < (\epsilon/(2k(1+\epsilon))) + 1$.

For any $i = 0, 1, \ldots, k-1$, let us consider an instance x of Π and the following non deterministic algorithm where p is the polynomial that bounds the value of all feasible solutions of Π:

- guess a candidate solution y with value at most $p(|x|)$;
- if $y \in \mathrm{Sol}_\Pi(x)$ and $m_\Pi(x, y) \leqslant r_n^{i+1} m(x, \mathtt{A}_\Pi(x))$, then return *yes*, otherwise return *no*.

Applying once again the technique of Theorem 1, we can construct k propositional formulæ $\varphi_0, \varphi_1, \ldots, \varphi_{k-1}$ such that for any truth assignment τ_i satisfying φ_i, $i = 0, 1, \ldots, k-1$, in polynomial time we can build a feasible solution y of the instance x with $m_\Pi(x, y) \geqslant r_n^i m(x, \mathtt{A}_\Pi(x))$.

Hence, the instance ψ of MAX 3-SAT that we consider is the following:

$$\psi = f(x, r) = \bigcup_{i=0}^{k-1} f_s(\varphi_i)$$

where f_s is the function defined in Lemma 1; w.l.o.g., we can suppose that all formulæ $f_s(\varphi_i)$, $i = 0, \ldots, k-1$, contain the same number of clauses.

Denote by T a satisfying truth assignment of ψ achieving approximation ratio r and by r_i the approximation ratio guaranteed by τ over $f_s(\varphi_i)$. By Lemma 2 we get:

$$\frac{m(r_i - 1)}{2r_i} \leqslant \mathrm{opt}(\psi) - m(\psi, T) \leqslant km\frac{r-1}{r}$$

Using this expression for $i = 0, \ldots, k-1$, we have $m(r_i - 1)/2r_i \leqslant km(r-1)/r$, which implies $1 - (2k(r-1)/r) \leqslant 1/r_i$ and, finally, $r_i \leqslant 1 + \epsilon$.

Using again Lemma 1, we derive that, for $i = 0, \ldots, k-1$, the truth assignment $\tau_i = g_s(\varphi_i, \tau)$ (where g_s is as defined in Lemma 1) satisfies φ_i if and only if φ_i is satisfiable. Let us call i^* the largest i for which τ_i satisfies φ_i; then:

$$r_n^{i^*} m(x, \mathtt{A}_\Pi(x)) \leqslant \mathrm{opt}_\Pi(x) \leqslant r_n^{i^*+1} m(x, \mathtt{A}_\Pi(x))$$

Starting from τ_{i^*}, we can then construct a solution y for Π whose value is at least $r_n^{i^*} m(x, \mathtt{A}_\Pi(x))$. This means that y guarantees an approximation ratio r_n. In other words, $r(x, y) \leqslant r_n = 1 + \alpha(r-1)$ and the reduction (f, g, α) that we have just defined (where g consists in applying g_s, determining i^* and constructing y starting from τ_{i^*}) is an AP-reduction.

Since Π is any maximization problem in **APX**, the completeness of MAX 3-SAT for the class of maximization problems in **APX** follows.

We now turn to the second part of the theorem. In fact, we still have to prove that all minimization problems in **APX** can be AP-reduced to maximization problems and, henceforth, to MAX 3-SAT.

Let us consider a minimization problem $\Pi \in \mathbf{APX}$ and an algorithm $\mathtt{A}$ with approximation ratio r for Π; let $k = \lceil r \rceil$. We can construct a maximization problem $\Pi' \in \mathbf{APX}$ and prove that $\Pi \leqslant_{\mathsf{AP}} \Pi'$. The two problems have the same instances and the same feasible solutions, while the objective function of Π' is defined as follows: given an instance x and a feasible solution y of x,

$$m_{\Pi'}(x,y) = \begin{cases} (k+1)m_\Pi(x,\mathtt{A}(x)) - km_\Pi(x,y) & \text{if } m_\Pi(x,y) \leqslant m_\Pi(x,\mathtt{A}(x)) \\ m_\Pi(x,\mathtt{A}(x)) & \text{otherwise} \end{cases}$$

Clearly, $m_\Pi(x,\mathtt{A}(x)) \leqslant \mathrm{opt}_{\Pi'}(x) \leqslant (k+1)m_\Pi(x,\mathtt{A}(x))$ and, by definition of Π', the algorithm $\mathtt{A}$ is also an approximation algorithm for this problem with approximation ratio $k+1$; therefore $\Pi' \in \mathbf{APX}$. The reduction from Π to Π' can now be defined as follows: for any instance x of Π, $f(x) = x$; for any instance x of Π and for any solution y of instance $f(x)$ of Π':

$$g(x,y) = \begin{cases} y & \text{if } m_\Pi(x,y) \leqslant m_\Pi(x,\mathtt{A}(x)) \\ \mathtt{A}(x) & \text{otherwise} \end{cases}$$

and $\alpha = k+1$. Note that f and g do not depend on the approximation ratio r.

We now show that the reduction we have just defined is an AP-reduction. Let y be an r'-approximate solution of $f(x)$; we have to show that the ratio $r_\Pi(x,g(x,y))$ of the solution $g(x,y)$ of the instance x of Π is smaller than, or equal to, $1+\alpha(r'-1)$. We have the following two cases: $m_\Pi(x,y) \leqslant m_\Pi(x,\mathtt{A}(x))$ and $m_\Pi(x,y) > m_\Pi(x,\mathtt{A}(x))$.

- In the case $m_\Pi(x,y) \leqslant m_\Pi(x,\mathtt{A}(x))$, we have: $g(x,y) = y$, $m_{\Pi'}(x,y) \geqslant \mathrm{opt}_{\Pi'}(x)/r$,

  $$m_\Pi(x,\mathtt{A}(x)) \leqslant \mathrm{opt}_{\Pi'}(x) \leqslant (k+1)m_\Pi(x,\mathtt{A}(x))$$

  and $m_\Pi(x,\mathtt{A}(x)) \leqslant \mathrm{opt}_\Pi(x)$. By considering such relations and the definition of α, we can derive:

  $$m_\Pi(x,y) \leqslant \left(1+\alpha\left(r'-1\right)\right)\mathrm{opt}_\Pi(x)$$

  In other words: $r_\Pi(x,g(x,y)) = r_\Pi(x,y) \leqslant 1+\alpha(r'-1)$.

- In the case $m_\Pi(x,y) > m_\Pi(x,\mathtt{A}(x))$, since $\alpha \geqslant 1$, we have:

  $$r_\Pi(x,g(x,y)) = r_\Pi(x,\mathtt{A}(x)) = r_{\Pi'}(x,y) \leqslant r' \leqslant 1+\alpha\left(r'-1\right)$$

In conclusion, all minimization problems in **APX** can be AP-reduced to maximization problems in **APX** and all maximization problems in **APX** can be AP-reduced to MAX 3-SAT. Since the AP-reduction is transitive the **APX**-completeness of MAX 3-SAT is proved. ∎

6.3 Negative results based on APX-completeness

Similarly to what we saw for completeness in **NPO**, also completeness in **APX** implies negative results in terms of approximability of optimization problems. In fact if we could prove that an **APX**-complete problem admits a PTAS, then so would all problems in **APX**. On the other side, it is well known that, unless $\mathbf{P} = \mathbf{NP}$ there are problems in **APX** that do not admit a PTAS (one example for all, MIN SCHEDULING ON IDENTICAL MACHINES, see [3]), therefore, under the same complexity theoretic hypothesis, no **APX**-complete problem admits a PTAS.

As a consequence of the results in the previous subsection, we can therefore assert that, unless $\mathbf{P} = \mathbf{NP}$, MAX 3-SAT does not admit a PTAS, neither do all other optimization problems that have been shown **APX**-complete: (MAX 2-SAT, MIN VERTEX COVER, MAX CUT, MIN METRIC TSP etc.).

Note that the inapproximability of MAX 3-SAT has been proved by Arora et al. ([1]) in a breakthrough paper by means of sophisticated techniques based on the concept of *probabilistically checkable proofs*, without any reference to the notion of **APX**-completeness. This fact, though, does not diminish the relevance of approximation preserving reductions and the related completeness notion. In fact most results that state the non existence of PTAS for **APX** optimization problems have been proved starting from MAX 3-SAT, via approximation preserving reductions that allow to carry over the inapproximability results from one problem to another. In second place, it is worth noting that the structure of approximation classes with respect to approximation preserving reductions is richer than it appears from this survey. For example, beside complete problems, other classes of problems can be defined inside approximation classes, identifying the so called *intermediate* problems (see [3]).

7 FT-reducibility

As we have already pointed out in Section 5, **PTAS**-completeness has been studied in [13] under the so-called F-reduction, preserving membership in **FPTAS**. Under this type of reducibility, a single problem, a rather artificial version of MAX WSAT has been shown **PTAS**-complete. In fact, F-reducibility is quite restrictive since it mainly preserves optimality, henceforth, existence of a **PTAS**-complete polynomially bounded problem is very unlikely.

In [8], a more "flexible" type of reducibility, called FT-reducibility has been introduced. It is formally defined as follows.

Definition 11. Let Π and Π' be two maximization integer-valued problems. Then, Π FT-reduces to Π' (denoted by $\Pi \leqslant_{\mathsf{FT}} \Pi'$) if, for any $\epsilon > 0$, there exist an oracle $\bigcirc_\alpha^{\Pi'}$ for Π' and an algorithm $\mathtt{A}_\epsilon$ calling $\bigcirc_\alpha^{\Pi'}$ such that:

- $\bigcirc_\alpha^{\Pi'}$ produces, for any $\alpha \in]0,1]$ and for any instance x' of Π', a feasible solution $\bigcirc_\alpha^{\Pi'}(x')$ of x' that is an $(1-\alpha)$-approximation;
- for any instance x of Π, $y = \mathtt{A}_\epsilon(\bigcirc_\alpha^{\Pi'}, x) \in \mathrm{Sol(x)}$; furthermore the approximation ratio of y is at least $(1-\epsilon)$;
- if $\bigcirc_\alpha^{\Pi'}(\cdot)$ runs in time polynomial in both $|f(x)|$ and $1/\alpha$, then $\mathtt{A}_\epsilon(\bigcirc_\alpha^{\Pi'}(f(x)), x)$ is polynomial in both $|x|$ and $1/\epsilon$. ∎

For the case where at least one among Π and Π' is a minimization problem it suffices to replace $1-\epsilon$ or/and $1-\alpha$ by $1+\epsilon$ or/and $1+\alpha$, respectively.

As one can see from Definition 11, FT-reduction is somewhat different from the other ones considered in this chapter and, in any case, it is not conformal to Definition 6. In fact, it resembles a Turing-reduction. Clearly, FT-reduction transforms a fully polynomial time approximation schema for Π' into a fully polynomial time approximation schema for Π, i.e., it preserves membership in **FPTAS**. Note also that the F-reduction, as it is defined in [13], is a special case of the FT-reduction, since the latter explicitly allows multiple calls to oracle $\bigcirc$ while for the former this fact is not explicit.

Theorem 4. *Let Π' be an **NP**-hard problem in **NPO**. If $\Pi' \in$ **NPO-PB** (the class of problems in **NPO** whose values are bounded by a polynomial in the size of the instance), then any **NPO** problem FT-reduces to Π'. Consequently, (i) $\overline{\textbf{PTAS}}^{\mathsf{FT}} =$ **NPO** and (ii) any **NP**-hard polynomially bounded problem in **PTAS** is **PTAS**-complete under FT-reductions.*

Sketch of proof. We first prove the following claim: *if an **NPO** problem Π' is **NP**-hard, then any **NPO** problem Turing-reduces (see [3]) to Π'.*

In order to prove this claim, let Π be an **NPO** problem and q be a polynomial such that $|y| \leqslant q(|x|)$, for any instance x of Π and for any feasible solution y of x. Assume that the encoding $n(y)$ of y is binary. Then $0 \leqslant n(y) \leqslant 2^{q(|x|)} - 1$. We consider problem $\hat{\Pi}$ which is the same as Π up to its value that is defined by:

$$m_{\hat{\Pi}}(x, y) = 2^{q(|x|)+1} m_{\Pi}(x, y) + n(y)$$

If $m_{\hat{\Pi}}(x, y_1) \geqslant m_{\hat{\Pi}}(x, y_2)$, then $m_{\Pi}(x, y_1) \geqslant m_{\Pi}(x, y_2)$. So, if a solution y is optimal for x, with respect to $\hat{\Pi}$, it is so with respect to Π. Remark now that $\hat{\Pi}$ and its evaluation version $\hat{\Pi}_e$ are equivalent since given the value of an optimal solution y, one can determine $n(y)$ (hence y) by computing the remainder of the division of this value by $2^{q(|x|)+1}$. Since Π' is **NP**-hard, it can be shown that one can solve the evaluation problem $\hat{\Pi}_e$, henceforth $\hat{\Pi}$ if one can solve, the (constructive) problem Π' and the claim is proved.

We now prove the following claim: *let* $\Pi' \in$ ***NPO-PB****; then, any* ***NPO*** *problem Turing-reducible to* Π' *is also* FT*-reducible to* Π'.

In order to prove this second claim, let Π be an **NPO** problem and suppose that there exists a Turing-reduction between Π and Π'. Let $\bigcirc_{\alpha}^{\Pi'}$ be as in Definition 11. Moreover, let p be a polynomial such that for any instance x' of Π' and for any feasible solution y' of x', $m(x', y') \leqslant p(|x'|)$. Let x be an instance of Π. The Turing-reduction claimed gives an algorithm solving Π using an oracle for Π'. Consider now this algorithm where we use, for any query to the oracle with the instance x' of Π', the approximate oracle $\bigcirc_{\alpha}^{\Pi'}(x')$, with $\alpha = 1/(p(|x'|) + 1)$. This algorithm is polynomial and produces an optimal solution, since a solution y' being an $(1 - (1/(p(|x'|) + 1)))$-approximation for x' is an optimal one. So, the claim is proved.

Combination of the above claims easily derives the theorem. ∎

Observe finally that MAX PLANAR INDEPENDENT SET and MIN PLANAR VERTEX COVER are in both **PTAS** ([7]) and **NPO-PB**. So, the following theorem concludes this section.

Theorem 5. MAX PLANAR INDEPENDENT SET *and* MIN PLANAR VERTEX COVER *are* ***PTAS****-complete under* FT*-reductions.*

8 Gadgets, reductions and inapproximability results

As it has been pointed out already in Section 3, in the context of approximation preserving reductions we call *gadget* a combinatorial structure which allows to transfer approximability (or inapproximability) results from a problem to another. A classical example is the set of ten 2-SAT clauses that we have used in Example 1 for constructing the 2-SAT formula starting from a 3-SAT formula. Although gadgets are used since the seminal work of Karp on reductions among combinatorial problems, the study of gadgets has been started in [9] and in [29]; from the latter derive most of the results discussed in this section.

In order to understand the role of gadgets in approximation preserving reductions, let us first go back to linear reductions and see what are the implications on the approximation ratio of two problems Π and Π', deriving from the fact that $\Pi \leqslant_{\mathsf{L}} \Pi'$. Suppose Π and Π' are minimization problems, f, g, α_1 and α_2 are the functions and constants that define the linear reduction, x is an instance of problem Π, $f(x)$ is the instance of problem Π' determined by the reduction, y is a solution of $f(x)$. Then, the following relationship holds between the approximation ratios of Π and Π':

$$r_{\Pi}(x, g(x, y)) \leqslant 1 + \alpha_1 \alpha_2 \left(r_{\Pi'}(f(x), y) - 1\right)$$

and, therefore, we have that:

$$r_{\Pi'} \leqslant 1 + \frac{r - 1}{\alpha_1 \alpha_2} \Longrightarrow r_{\Pi} \leqslant r$$

In the case of maximization problems we have, instead, the following implication:

$$r_{\Pi'} = \frac{r\alpha_1\alpha_2}{r\left(\alpha_1\alpha_2 - 1\right) + 1} \Longrightarrow r_{\Pi} = r$$

In the particular case of the reduction between MAX 3-SAT and MAX 2-SAT, we have $\alpha_1\alpha_2 = 13$ and, therefore, we can infer the following results on the approximability upper bounds and lower bounds of the two problems, which may be proved by a simple calculation:

- since it is known that MAX 2-SAT can be approximated with the ratio 0.931 ([15]), then MAX 3-SAT can be approximated with ratio 0.103;
- Since it is known that MAX 3-SAT cannot be approximated beyond the threshold 7/8, then MAX 2-SAT cannot be approximated beyond the threshold 103/104.

Although better bounds are now known for these problems (see Karloff and Zwick [18]), it is important to observe that the above given bounds may be straightforwardly derived from the linear reduction between the two problems and are useful to show the role of gadgets. In such reduction, the structure of the gadget is crucial (it determines the value α_1) and it is clear that better bounds could be achieved if the reduction could make use of "smaller" gadgets. In fact, in [29], by cleverly constructing a more sophisticated type of gadget (in which, in particular, clauses have real weights), the authors derive a 0.801 approximation algorithm for MAX 3-SAT, improving on previously known bounds.

Based on [9], in [29] the notion of α-gadget (that is, gadget with performance α) is abstracted and formalized with reference to reductions among constraint satisfaction problems. In the same paper, it is shown that, under suitable circumstances, the search for (possibly optimum) gadgets to be used in approximation preserving reductions, can be pursued in a systematic way by means of a computer program. An example of the results that may be achieved in this way is the following.

Let PC_0 and PC_1 be the families of constraints over three binary variables defined as:

$$\mathrm{PC}_i(a, b, c) = \begin{cases} 1 & \text{if } a \oplus b \oplus c = i \\ 0 & \text{otherwise} \end{cases}$$

and let DICUT be the family of constraints corresponding to directed cuts in a graph. There exists optimum 6.5 gadgets (automatically derived by the computer program) reducing PC_0 and PC_1 to DICUT. As a consequence, for any $\epsilon > 0$, MAX DICUT is hard to approximate to within $12/13 + \epsilon$.

9 Conclusion

A large number of other approximation preserving reductions among optimization problems, beside those introduced in this chapter, have been introduced throughout the years. Here we have reported only the major developments. Other overviews of the world of approximation preserving reductions can be found in [11] and [12].

As we have already pointed out in Section 2, we have not dealt in this survey with approximability classes beyond **APX**, even if intensive studies have been performed, mainly for **Poly-APX**. In [20], completeness results are established, under the E-reduction, for **Poly-APX-PB** (the class of problems in **Poly-APX** whose values are bounded by a polynomial in the size of the instance). Indeed, as we have already discussed in Section 5, use of restrictive reductions as the E-reducibility, where the functions f and g do not depend on any parameter ϵ seems very unlikely to be able to handle **Poly-APX**-completeness. As it is shown in [8], completeness for the whole **Poly-APX** can be handled, for instance, by using PTAS-reduction, a further

relaxation of the AP-reduction where the dependence between the approximation ratios of Π and Π' is not restricted to be linear ([14]). Under PTAS-reduction, MAX INDEPENDENT SET is **Poly-APX**-complete ([8]).

Before concluding, it is worth noting that a structural development (based on the definition of approximability classes, approximation preserving reductions and completeness results), analogous to the one that has been carried on for the classical approach to the theory of approximation, has been elaborated also for the differential approach (see [2, 8]). In [2] and in [8] the approximability classes **DAPX**, **Poly-DAPX** and **DPTAS** are introduced, suitable approximation preserving reductions are defined and complete problems in **NPO**, **DAPX**, **Poly-DAPX** and **DPTAS**, under such kind of reductions, are shown.

References

[1] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and intractability of approximation problems. In *Proc. FOCS'92*, pages 14–23, 1992.

[2] G. Ausiello, C. Bazgan, M. Demange, and V. Th. Paschos. Completeness in differential approximation classes. *International Journal of Foundations of Computer Science*. To appear.

[3] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, and M. Protasi. *Complexity and approximation. Combinatorial optimization problems and their approximability properties*. Springer, Berlin, 1999.

[4] G. Ausiello, P. Crescenzi, and M. Protasi. Approximate solutions of NP optimization problems. *Theoret. Comput. Sci.*, 150:1–55, 1995.

[5] G. Ausiello, A. D'Atri, and M. Protasi. Structure preserving reductions among convex optimization problems. *J. Comput. System Sci.*, 21:136–153, 1980.

[6] G. Ausiello, A. D'Atri, and M. Protasi. Lattice-theoretical ordering properties for NP-complete optimization problems. *Fundamenta Informaticæ*, 4:83–94, 1981.

[7] B. S. Baker. Approximation algorithms for NP-complete problems on planar graphs. *J. Assoc. Comput. Mach.*, 41(1):153–180, 1994.

[8] C. Bazgan, B. Escoffier, and V. Th. Paschos. Completeness in standard and differential approximation classes: Poly-(D)APX- and (D)PTAS-completeness. *Theoret. Comput. Sci.*, 339:272–292, 2005.

[9] M. Bellare, O. Goldreich, and M. Sudan. Free bits and non-approximability — towards tight results. *SIAM J. Comput.*, 27(3):804–915, 1998.

[10] S. A. Cook. The complexity of theorem-proving procedures. In *Proc. STOC'71*, pages 151–158, 1971.

[11] P. Crescenzi. A short guide to approximation preserving reductions. In *Proc. Conference on Computational Complexity*, pages 262–273, 1997.

[12] P. Crescenzi, V. Kann, R. Silvestri, and L. Trevisan. Structure in approximation classes. *SIAM J. Comput.*, 28(5):1759–1782, 1999.

[13] P. Crescenzi and A. Panconesi. Completeness in approximation classes. *Information and Computation*, 93(2):241–262, 1991.

[14] P. Crescenzi and L. Trevisan. On approximation scheme preserving reducibility and its applications. *Theory of Computing Systems*, 33(1):1–16, 2000.

[15] U. Feige and M. X. Goemans. Approximating the value of two prover proof systems, with applications to max 2sat and max dicut. In *Proc. Israel Symposium on Theory of Computing and Systems, ISTCS'95*, pages 182–189, 1995.

[16] M. R. Garey and D. S. Johnson. *Computers and intractability. A guide to the theory of NP-completeness*. W. H. Freeman, San Francisco, 1979.

[17] D. S. Johnson. Approximation algorithms for combinatorial problems. *J. Comput. System Sci.*, 9:256–278, 1974.

[18] H. Karloff and U. Zwick. A 7/8-approximation for MAX 3SAT? In *Proc. FOCS'97*, pages 406–415, 1997.

[19] R. M. Karp. Reducibility among combinatorial problems. In R. E. Miller and J. W. Thatcher, editors, *Complexity of computer computations*, pages 85–103. Plenum Press, New York, 1972.

[20] S. Khanna, R. Motwani, M. Sudan, and U. Vazirani. On syntactic versus computational views of approximability. *SIAM J. Comput.*, 28:164–191, 1998.

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

[22] P. Orponen and H. Mannila. On approximation preserving reductions: complete problems and robust measures. Technical Report C-1987-28, Dept. of Computer Science, University of Helsinki, Finland, 1987.

[23] C. H. Papadimitriou. *Computational complexity*. Addison-Wesley, 1994.

[24] C. H. Papadimitriou and M. Yannakakis. Optimization, approximation and complexity classes. *J. Comput. System Sci.*, 43:425–440, 1991.

[25] V. Th. Paschos. *Complexité et approximation polynomiale*. Hermès, Paris, 2004.

[26] A. Paz and S. Moran. Non deterministic polynomial optimization problems and their approximations. *Theoret. Comput. Sci.*, 15:251–277, 1981.

[27] H. U. Simon. Continuous reductions among combinatorial optimization problems. *Acta Informatica*, 26:771–785, 1989.

[28] H. U. Simon. On approximate solutions for combinatorial optimization problems. *SIAM J. Disc. Math.*, 3(2):294–310, 1990.

[29] L. Trevisan, G. B. Sorkin, M. Sudan, and D. P. Williamson. Gadgets, approximation, and linear programming. *SIAM J. Comput.*, 29(6):2074–2097, 2000.