Submitted to
Theoretical Computer Science

On Approximation Scheme Preserving Reducibility and its Applications[⋆]

Pierluigi Crescenzi and Luca Trevisan

Dipartimento di Scienze dell'Informazione
Università degli Studi di Roma "La Sapienza"
Via Salaria 113, 00198 Roma, Italy
E-mail: {`piluc,trevisan`}`@dsi.uniroma1.it`.

April 28, 1995

Summary. In this paper we generalize the notion of polynomial-time approximation scheme preserving reducibility, called PTAS-reducibility, introduced in a previous paper. As a first application of this generalization, we prove the APX-completeness of a polynomially bounded optimization problem, that is, an APX problem whose measure function is bounded by a polynomial in the length of the instance and such that any APX problem is reducible to it. As far as we know, no such problem was known before. This result has been recently used to show that several natural optimization problem are APX-complete, such as MAX CUT, MAX SAT, MIN NODE COVER, and MIN Δ-TSP.

Successively, we apply the notion of APX-completeness to the study of the relative complexity of evaluating an r-approximate value and computing an r-approximate solution for any r. We first show that if $P \neq NP \cap coNP$ then the former question can be easier than the latter even if the optimization problem is NP-hard. We therefore give strong evidence that if an optimization problem is APX-complete then the two questions are both hard.

1. Introduction

It is well known that for several important optimization problems, such as the traveling salesman problem or the graph coloring problem, determining an optimal solution is extremely time consuming due to the inherent complexity of such problems. For this reason when we have to solve problems of this kind we must restrict ourselves to compute approximate solutions, that is, solutions whose performance ratio is guaranteed to be bounded by a constant [10].

In this paper, we focus our attention on the two classes APX and PTAS, that is, the class of problems that are approximable within a factor r for a given r and the class of problems that admit an approximation scheme, that is, are approximable within any factor r. It is well-known that PTAS is strictly contained in APX if and only if $P \neq NP$.

Several notions of approximation scheme preserving reducibilities have been introduced in [5, 14, 15] with the aim of establishing hardness and completeness results in APX and of

[⋆] Research partially supported by the MURST project *Efficienza di Algoritmi e Progetto di Strutture Informative.*

deriving proofs of intractability of arbitrary approximation from them (see also Chap. 3 of [11] for a survey on the notion of reducibility among optimization problems). In particular, in [5] an approximation scheme preserving reducibility, called PTAS-reducibility, was defined and the existence of APX-complete problems was shown. Independently, a more restricted kind of reducibility, called L-reducibility, was introduced in [15] and several completeness results for a subclass of APX were proved. In Sect. 2 we generalize the PTAS-reducibility and prove the existence of polynomially bounded APX-complete problems, that is, APX-complete problems whose measure function is bounded by a polynomial in the length of the input. As far as we know, this is the first example of such problems. This result has been recently used in [12] in order to prove the APX-completeness of several natural optimization problems, such as Max Cut, Max Sat, Min Node Cover, and Min Δ-TSP.

Successively, we apply the notion of APX-completeness to the study of the relative complexity of evaluating an r-approximate value and computing an r-approximate solution for any r. The relative complexity of checking and evaluating a function was first considered in [17]. It is well-known, for example, that checking whether an array is already sorted is simpler than sorting it. Valiant proved that, indeed, the two questions are equivalent if and only if $\mathrm{P} = \mathrm{NP}$. In [16] and, successively, in [6] the relative complexity of evaluating the optimum measure and constructing an optimum solution for optimization problems was analyzed. For example, we can either compute the size of a maximum clique in a given graph or list the nodes of a maximum clique. Crescenzi and Silvestri gave strong evidence that the latter question may be harder than the former even though it was known that the two questions are equivalent whenever the optimization problem is NP-hard.

In Sect. 3 we show that if $\mathrm{P} \neq \mathrm{NP} \cap co\mathrm{NP}$ then a problem exists in APX $-$ PTAS whose optimum measure can be approximated within any factor. Moreover, this problem is NP-hard. Thus the property of NP-hardness is not sufficient to guarantee the equivalence between constructive and non-constructive approximation and a different notion of completeness seems to be required. Indeed, we show that no APX-complete problem admits a non-constructive polynomial-time approximation scheme, unless $\mathrm{NP} = co\mathrm{NP}$.

Finally, in Sect. 4, in order to strengthen the above result we characterize the class of problems that admit non-constructive polynomial-time approximation schemes in terms of specific classes of languages and of a complexity class raised in the recent theory of parameterized complexity.

1.1. Preliminaries

The basic ingredients of an optimization problem are the set of instances or input objects, the set of feasible solutions or output objects associated to any instance, and the measure defined for any feasible solution. We thus give the following definition.

Definition 1. *An* NPO problem *A is a fourtuple $(I, sol, m, goal)$ such that*

1. *I is the set of the* instances *of A and it is recognizable in polynomial time.*
2. *Given an instance x of I, $sol(x)$ denotes the set of* feasible solutions *of x. These solutions are short, that is, a polynomial p exists such that, for any $y \in sol(x)$, $|y| \leq p(|x|)$. Moreover, it is decidable in polynomial time whether, for any x and for any y such that $|y| \leq p(|x|)$, $y \in sol(x)$.*
3. *Given an instance x and a feasible solution y of x, $m(x, y)$ denotes the positive integer* measure *of y and is computable in polynomial time.*

4. goal $\in \{\max, \min\}$.

If a polynomial q *exists such that, for any instance* x *and for any solution* y *of* x, $m(x,y) \leq q(|x|)$, *then* A *is said to be* polynomially bounded.

The *class* NPO is the set of all NPO problems.

The goal of an NPO problem with respect to an instance x is to find an *optimum solution*, that is, a feasible solution y such that

$$m(x,y) = goal\{m(x,y') \ : \ y' \in sol(x)\} \ .$$

The following are some well-known examples of NPO problems. For the sake of simplicity, we avoid to specify the goal since it can be immediately derived from the name of the problem itself.

Max Cut

Instance: Graph $G = (V,E)$ and integer weight $w(e)$ for each $e \in E$.
Solution: Subset $V' \subseteq V$.
Measure: Sum of the weights of the edges from E that have one endpoint in V' and one endpoint in $V - V'$.

Max Knapsack

Instance: Finite set $I = \{1,\ldots,n\}$, for each $i \in I$ an integer size a_i and an integer profit p_i, and integer b.
Solution: Function $f : I \rightarrow \{0,1\}$ such that $\sum_{i \in I} f(i)a_i \leq b$.
Measure: $\sum_{i \in I} f(i)p_i$.

Max Sat

Instance: Set U of variables and collection C of clauses over U.
Solution: Truth assignment to the variables in U.
Measure: Number of satisfied clauses.

Min Node Cover

Instance: Graph $G = (V,E)$.
Solution: Subset $V' \subseteq V$ such that, for each edge $(u,v) \in E$, at least one of u and v belongs to V'.
Measure: $|V'|$.

Min Δ-TSP

Instance: Complete weighted graph $G = (V, E)$ such that the triangle inequality is satisfied.
Solution: A permutation π of V.
Measure: $\sum_{i=1}^{|V|-1} w(v_{\pi(i)}, v_{\pi(i+1)}) + w(v_{\pi(|V|)}, v_{\pi(1)})$ where $w(e)$ is the weight of edge e.

In the following *sol** will denote the multi-valued function mapping an instance x to the set of optimum solutions, while *opt* will denote the function mapping an instance x to the measure of an optimum solution. Moreover, in this paper we will focus our attention on maximization problems only so that we will not specify the goal of the problem.

Definition 2. *Given an* NPO *problem A, the* language L_A associated with *A is defined as*

$$L_A = \{(x, k) : x \in I \wedge opt(x) \geq k\} \ .$$

Whenever L_A is NP*-complete, A is said to be* NP*-hard.*

It is well-known that if P $\neq$ NP, then no NP-hard NPO problem is solvable in polynomial time. In these cases we sacrifice optimality and start looking for approximate solutions computable in polynomial time.

Definition 3. *Let A be an* NPO *problem. Given an instance x and a feasible solution y of x, the* performance ratio of y (with respect to x) *is defined as*

$$R(x, y) = \frac{opt(x)}{m(x, y)} \ .$$

The performance ratio is always a number greater than 1 and is as close to 1 as the feasible solution is close to the optimum one.

Definition 4. *An* NPO *problem A belongs to the* class APX *if a rational $r > 1$ and a polynomial-time algorithm T exist such that, for any instance x, the performance ratio of the feasible solution $T(x)$ is at most r.*

Definition 5. *An* NPO *problem A belongs to the* class PTAS *if it admits a* polynomial-time approximation scheme, *that is, an algorithm T such that, for any instance x of A and for any rational $r > 1$, $T(x, r)$ returns a feasible solution whose performance ratio is at most r in time bounded by $q_r(|x|)$ where q_r is a polynomial.*

Clearly, PTAS $\subseteq$ APX. It is also well-known that this containment is strict if and only if P $\neq$ NP. An extensive survey of results on these two classes is contained in [2].

Definition 6. *An* NPO *problem A belongs to the* class *nc*PTAS *if it admits a polynomial-time* non-constructive *approximation scheme, that is, an algorithm T such that, for any instance x of A and for any rational $r > 1$, $T(x, r)$ returns a value between $(1/r)opt(x)$ and $opt(x)$ in time bounded by $q_r(|x|)$ where q_r is a polynomial. A value between $(1/r)opt(x)$ and $opt(x)$ is also said an r-*approximate value for *x.*

Observe that the time complexity of a (non-constructive) approximation scheme in the last two definitions may be exponential in the rational $1/(r-1)$, that is, it may be of the type $2^{1/(r-1)}p(|x|)$ or $|x|^{1/(r-1)}$ where p is a polynomial.

2. PTAS-Reducibility and APX-Completeness

The many-to-one polynomial-time reducibility is clearly inadequate to study the approximability properties of optimization problems. Indeed, if we want to map an optimization problem A into an optimization problem B then not only do we need a function mapping instances of A into instances of B but also a function mapping back solutions of B into solutions of A preserving the performance ratio.

Definition 7. *Let A and B be two* NPO *problems. A is said to be* PTAS-reducible *to B, in symbols $A \leq B$, if three computable functions f, g, and c exist such that:*

1. *For any $x \in I_A$ and for any $r > 1$, $f(x,r) \in I_B$ is computable in time polynomial with respect to $|x|$.*
2. *For any $x \in I_A$, for any $r > 1$, and for any $y \in sol_B(f(x,r))$, $g(x,y,r) \in sol_A(x)$ is computable in time polynomial with respect to both $|x|$ and $|y|$.*
3. *$c : (1,\infty) \to (1,\infty)$.*
4. *For any $x \in I_A$, for any $r > 1$, and for any $y \in sol_B(f(x,r))$,*

$$R_B(f(x,r),y) \leq c(r) \textit{ implies } R_A(x,g(x,y,r)) \leq r \ .$$

The triple (f,g,c) is said to be a PTAS-reduction *from A to B.*

Remark 1. The PTAS-reducibility is a generalization of a reducibility introduced in [5] and called P-reducibility. Indeed, the only difference between the PTAS-reducibility and the P-reducibility is the fact that f and g may depend on r. Moreover, in [15] a different kind of reducibility between optimization problems was defined which is a restriction of the P-reducibility and is called L-reducibility. Indeed, an L-reduction turns out to be a P-reduction with $c(r) = \frac{\alpha\beta}{1/r+\alpha\beta-1}$ where α and β are constants. In [4] a new reducibility called AP-reduction is introduced: an AP-reduction is essentially a PTAS-reduction where function c is linear. The main result of this paper, that is, the existence of a polynomially bounded APX-complete problem, also holds with respect to the AP-reducibility, and there is a certain evidence that this result cannot be proved with respect to any stricter reducibility than the AP-reducibility (see [4]).

Proposition 1. *If $A \leq B$ and $B \in$* PTAS*, then $A \in$* PTAS*.*

Proof. Let T_B be a polynomial-time approximation scheme for B and let (f,g,c) be a PTAS-reduction from A to B. Then

$$T_A(x,r) = g(x, T_B(f(x,r), c(r)), r)$$

is a polynomial-time approximation scheme for A. □

Definition 8. *An* NPO *problem A in* APX *is* APX*-complete if, for any other problem B in* APX*, $B \leq A$.*

In [14] a reducibility slightly stronger than the PTAS-reducibility was defined but no APX-completeness result was proved. The following problem, called Max Bounded Weighted Sat or, simply, MBWS, has been instead shown to be APX-complete in [5].

1. An instance is a Boolean formula in conjunctive normal form (in short, CNF-formula) φ with variables $x_1, \ldots, x_n$ of weights $w_1, \ldots, w_n$ such that

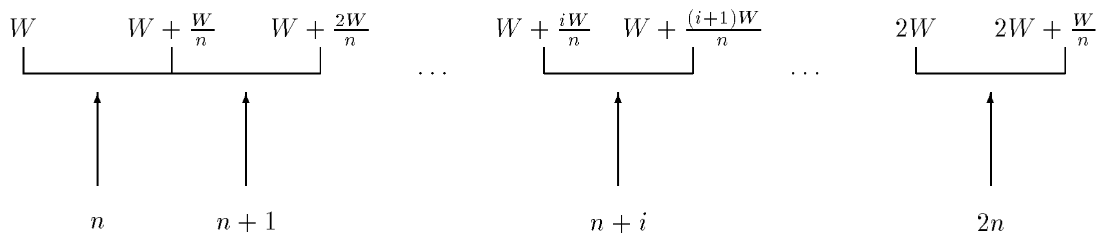
$$W \leq \sum_{i=1}^{n} w_i \leq 2W \ ,$$

where W is an integer.

2. For any CNF-formula φ, a feasible solution is a truth assignment to the variables.
3. For any CNF-formula φ and for any truth assignment τ,

$$m(\varphi,\tau) = \begin{cases} \max(W, \sum_{i=1}^{n} w_i \tau(x_i)) & \text{if } \tau \text{ satisfies } \varphi, \\ W & \text{otherwise.} \end{cases}$$

Let us now consider a polynomially bounded version of the above problem, called Max Polynomially Bounded Weighted Sat or, simply, MPBWS. Intuitively, this version is obtained by splitting the interval $[W, 2W + W/n)$, that is, an interval containing all possible measures of solutions of MBWS, into n intervals $[W + iW/n, W + (i+1)W/n)$ for $i = 0, \ldots, n$ such that any solution in the ith interval is assigned a new measure equal to $n + i$ as shown below.



Formally, MPBWS is equal to MBWS apart from the measure function which is defined as follows

$$m_{\text{MPBWS}}(x,\tau) = n + \left\lfloor \frac{n(m_{\text{MBWS}}(x,\tau) - W)}{W} \right\rfloor$$

where m_{MBWS} and m_{MPBWS} denote the measure functions of MBWS and MPBWS, respectively (this 'scaled' version of MPBWS was first suggested in [12]). Observe that according to the above definition, for any instance x of MPBWS and for any truth-assignment τ, $m_{\text{MPBWS}}(x,\tau) \leq 2n$, that is, this problem is indeed polynomially bounded.

Theorem 1. MBWS *is* PTAS-*reducible to* MPBWS.

Proof. Let $x = (\varphi, w_1, \ldots, w_n, W)$ denote an instance of MBWS. The reduction is then defined as follows (note that we are using the fact that g may depend on r).

1. For any $r > 1$, $f(x,r) = x$.
2. For any τ and for any $r > 1$,

$$g(x,\tau,r) = \begin{cases} \tau & \text{if } r > 1 + 2/n, \\ \tau^* & \text{otherwise} \end{cases}$$

where τ^* denotes an optimum solution for MBWS.

3. For any $r > 1$, $c(r) = (r+1)/2$.

Observe that, according to the definition of the PTAS-reducibility, the running time of g can be exponential in $1/(r-1)$. If $r \leq 1+2/n$ then g has enough time to compute τ^* so that, in this case, the fourth condition in the definition of PTAS-reducibility is clearly satisfied.

Assume now that $r > 1+2/n$, and let x be an instance and τ be a solution such that $R_{\mathrm{MPBWS}}(x,\tau) \leq c(r)$. Moreover, let

$$i_\tau = \left\lfloor \frac{n(m_{\mathrm{MBWS}}(x,\tau) - W)}{W} \right\rfloor \ .$$

Then

$$R_{\mathrm{MPBWS}}(x,\tau) = \frac{n + i_{\tau^*}}{n + i_\tau}$$

and

$$\begin{aligned} R_{\mathrm{MBWS}}(x,\tau) &= \frac{opt_{\mathrm{MBWS}}(x)}{m_{\mathrm{MBWS}}(x,\tau)} \\ &\leq \frac{W(i_{\tau^*}/n + 1 + 1/n)}{W(i_\tau/n+1)} \\ &= \frac{(i_{\tau^*} + n + 1)}{i_\tau + n} \\ &= R_{\mathrm{MPBWS}}(x,\tau) + \frac{1}{i_\tau + n} \\ &\leq R_{\mathrm{MPBWS}}(x,\tau) + 1/n \\ &\leq (r+1)/2 + (r-1)/2 = r \ . \end{aligned}$$

We thus have that, in both cases, the fourth condition in the definition of PTAS-reducibility is satisfied and this concludes the proof. □

Corollary 1. MPBWS *is* APX-*complete.*

Proof. It follows from the APX-completeness of MBWS and from the above theorem. □

As far as we know, this is the first example of a polynomially bounded APX-complete problem. By using this result, in [12] several other important problems are shown to be APX-complete, such as Max Cut, Max Sat, Min Node Cover, and Min Δ-TSP. Thus, each of these problems is the hardest within APX and, as already known, does not admit a polynomial-time approximation scheme unless P = NP.

2.1. Comparison with Other Reducibilities

By making use of either the P-reducibility [5], or the L-reducibility [15], or the E-reducibility [12] it does not seem possible to prove the APX-completeness of MPBWS and thus of the other natural problems. The difficulty is mainly due to the fact that these reducibilities do not allow the function g to depend on r: as a consequence, this function is forced to map optimum solutions into optimum solutions. The following definitions allow to formally capture the property of a reducibility to be optimum-preserving.

Definition 9. *Given an* NPO *problem A, a language L belongs to* P^A *if and only if two polynomial-time computable functions f and g exist such that, for any x, $f(x)$ is an instance of A, and, for any $y \in sol^*(f(x))$, $g(x,y)=1$ if and only if $x \in L$.*

Definition 10. *Let $\leq$ be a preorder in a set $\mathcal{C}$, that is, a transitive reflexive binary relation. If $\mathcal{A} \subseteq \mathcal{C}$ and, for any $b \in \mathcal{A}$, $b \leq a$ with $a \in \mathcal{A}$, then a is said to be a* $\leq$-maximum in *$\mathcal{A}$.*

Definition 11. *Let $\leq$ be a preorder in* NPO, *then $\leq$ is said to be* optimum preserving *if $A \leq B$ implies* $\mathrm{P}^A \subseteq \mathrm{P}^B$.

It is easy to see that the L-reducibility, the E-reducibility, and the P-reducibility are optimum preserving preorders in NPO, and that a $\leq$-maximum in APX is indeed an APX-complete problem.

Definition 12. *A language L belongs to the* class $\mathrm{P}^{\mathrm{NP}[f(n)]}$ *if and only if it is decidable by a polynomial-time oracle Turing machine which asks at most $f(n)$ queries to an* NP-*complete oracle, where n is the input size. Moreover, if F is a class of functions, then* $\mathrm{P}^{\mathrm{NP}[F]} = \bigcup_{f \in F} \mathrm{P}^{\mathrm{NP}[f(n)]}$. *Finally, let* $\mathrm{P}^{\mathrm{NP}} = \mathrm{P}^{\mathrm{NP}[n^{O(1)}]}$.

Proposition 2. *If an optimum preserving preorder $\leq$ exists in* NPO *such that a polynomially bounded problem A is a $\leq$-maximum in* APX *then* $\mathrm{P}^{\mathrm{NP}} = \mathrm{P}^{\mathrm{NP}[O(\log n)]}$.

Proof. Recall that from the results of [13] it follows that $\mathrm{P}^{\mathrm{NP}} \subseteq \mathrm{P}^{\mathrm{MK}}$, where MK stands for Max Knapsack, moreover it is well-known that Max Knapsack belongs to PTAS, and then to APX [9]. If A is a $\leq$-maximum in APX then Max Knapsack $\leq A$, so that $\mathrm{P}^{\mathrm{NP}} \subseteq \mathrm{P}^{\mathrm{MK}} \subseteq \mathrm{P}^A$ (since $\leq$ is optimum preserving). We will now show that $\mathrm{P}^A \subseteq \mathrm{P}^{\mathrm{NP}[O(\log n)]}$.

Let L be a language in P^A and let f and g be the functions witnessing that $L \in \mathrm{P}^A$. Since A is polynomially bounded and f is computable in polynomial time, then a polynomial q exists such that, for any x, $opt(f(x)) \leq q(|x|)$. By applying the binary search technique, it is easy to see that, for any x, $opt(f(x))$ is computable in polynomial time by asking at most $\lceil \log q(|x|) \rceil$ queries to an NP-complete oracle. Moreover, we have that, for any x, x belongs to L if and only if a solution y for $f(x)$ exists such that $m(f(x),y) = opt(f(x))$ and $g(x,y)=1$. That is, given the value of $opt(f(x))$, deciding whether $x \in L$ is an NP problem, and then $L \in \mathrm{P}^{\mathrm{NP}[\lceil \log q(n) \rceil + 1]}$.

In conclusion, $\mathrm{P}^{\mathrm{NP}} \subseteq \mathrm{P}^A \subseteq \mathrm{P}^{\mathrm{NP}[O(\log n)]}$. □

Corollary 2. Max Cut, Max Satisfiability, Min Δ-TSP, Min Node Cover *are not* APX-*complete with respect to the L-reducibility, or with respect to the E-reducibility, or with respect to the P-reducibility, unless* $\mathrm{P}^{\mathrm{NP}} = \mathrm{P}^{\mathrm{NP}[O(\log n)]}$.

The possibility that $\mathrm{P}^{\mathrm{NP}} = \mathrm{P}^{\mathrm{NP}[O(\log n)]}$ seems to be quite unlikely and, even if this event is currently not yet related to any other collapse in structural complexity theory, it is conjectured that it implies the collapse of the polynomial-time hierarchy [8, 18].

3. Evaluating, Constructing, and APX-Completeness

In this section we shall see that APX-complete problems not only do not belong to PTAS but they do not admit a non-constructive polynomial-time approximation scheme either (unless NP = *co*NP). Actually, one could think that this is true for any NP-hard optimization problem since for these problems it is known that evaluating the optimum measure and constructing an optimum solution are computationally equivalent [16]. The following result states that this intuition is wrong.

Theorem 2. *If* $\mathrm{P} \neq \mathrm{NP} \cap co\mathrm{NP}$ *then an* NP*-hard NPO problem exist that belongs to* $\mathrm{APX} \cap nc\mathrm{PTAS} - \mathrm{PTAS}$.

Proof. Assume that $\mathrm{P} \neq \mathrm{NP} \cap co\mathrm{NP}$. Let $L \in \mathrm{NP} \cap co\mathrm{NP} - \mathrm{P}$ and let NT and NT^c be the non-deterministic Turing machines deciding L and L^c in polynomial time, respectively. We then define the NPO problem A as follows.

1. I contains pairs (x, φ) where x is an instance of L and φ is a CNF-formula.
2. For any pair (x, φ), a feasible solution is a pair (y, τ) where y is either a computation path of $NT(x)$ or a computation path of $NT^c(x)$ and τ is a truth-assignment for φ.
3. For any instance (x, φ) of length n and for any feasible solution (y, τ), the measure function is defined as

$$m((x,\varphi),(y,\tau)) = \begin{cases} n/2 & \text{if } y \text{ is a rejecting computation,} \\ n & \text{if } y \text{ is an accepting computation} \\ & \text{and } \tau \text{ does not satisfy } \varphi, \\ n+1 & \text{otherwise.} \end{cases}$$

Clearly, A belongs to APX and, for any CNF-formula φ, φ is satisfiable if and only if $((x_{\text{yes}}, \varphi), n+1) \in L_A$ where x_{yes} is any word in L and n is equal to the length of $(x_{\text{yes}}, \varphi)$. Thus A is NP-hard.

It is also easy to see that $A \in nc\mathrm{PTAS}$. Indeed, the following algorithm is a non-constructive polynomial-time approximation scheme: for any instance (x, φ) and for any $r > 1$,

$$T((x,\varphi),r) = \begin{cases} n & \text{if } r \geq (n+1)/n \text{ or} \\ & r < (n+1)/n \text{ and } \varphi \text{ is not satisfiable,} \\ n+1 & \text{if } r < (n+1)/n \text{ and } \varphi \text{ is satisfiable} \end{cases}$$

where n denotes the length of the instance.

Finally, suppose that A admits a polynomial-time approximation scheme T and, for any instance (x, φ), consider the feasible solution $(y, \tau) = T((x, \varphi), 3/2)$. Then y must be an accepting computation path and $x \in L$ if and only if y is a computation path of $NT(x)$. That is, L belongs to P contradicting the hypothesis. In conclusion we have proved that $A \in \mathrm{APX} \cap nc\mathrm{PTAS} - \mathrm{PTAS}$. □

Hence, the notion of NP-hardness is not sufficient to guarantee the equivalence between non-constructive and constructive approximation schemes. We shall now see that, whenever an optimization problem is APX-complete, both evaluating r-approximate values and computing r-approximate solutions for any r are computationally hard. To this aim, let us first recall a result obtained in [1].

Theorem 3 ([1]). *Let L be a language in* NP. *Then a rational $r > 1$ exists such that, for any x, a CNF-formula φ_x is computable in polynomial time for which the following hold.*

1. If $x \in L$ then φ_x is satisfiable.
2. If $x \notin L$ then any truth-assignment satisfies less than a fraction $1/r$ of all the clauses of φ_x.

Theorem 4. *An* APX*-complete problem A exists that admits a non-constructive polynomial-time approximation scheme if and only if* $\mathrm{NP} = \mathrm{coNP}$.

Proof. Let A be an APX-complete problem and let (f, g, c) be a PTAS-reduction from Max Sat to A. Moreover, let L be an NP-complete language. From Theorem 3, it follows that, for any x, the CNF-formula φ_x satisfies the following two implications: (a) if $x \in L$ then

$opt_{\mathrm{MSAT}}(\varphi_x) = m$ and (b) if $x \notin L$ then $opt_{\mathrm{MSAT}}(\varphi_x) < (1/r)m$ where m denotes the number of clauses of φ_x. From the definition of PTAS-reducibility it follows also that if y is a feasible solution of $f(\varphi_x, r)$ whose performance ratio is at most $c(r)$, then $\tau_y = g(\varphi_x, y, r)$ is a truth assignment whose performance ratio is at most r.

It is then easy to verify that τ_y satisfies less than $(1/r)m$ clauses of φ_x if and only if $x \notin L$. Indeed, if $x \notin L$ then any truth assignment satisfies less than $(1/r)m$ clauses. Conversely, if τ_y satisfies less than $(1/r)m$ clauses then $opt_{\mathrm{MSAT}}(\varphi_x) \leq m_{\mathrm{MSAT}}(\varphi_x, \tau_y)r < m$, that is, $x \notin L$.

If A admits a non-constructive polynomial-time approximation scheme T, then we can develop a polynomial-time non-deterministic algorithm to decide the complement of L. Such an algorithm is shown in Fig. 1.

```
begin
  compute $\varphi_x$;
  $m$:= number of clauses of $\varphi_x$;
  $a$:= $T(f(\varphi_x, r), c(r))$;
  guess $y$ feasible solution of $f(\varphi_x, r)$;
  if $m_A(f(\varphi_x, r), y) < a$ then reject
  else begin
    $\tau_y$:= $g(\varphi_x, y, r)$;
    if $m_{\mathrm{MSAT}}(\varphi_x, \tau_y) < (1/r)m$ then accept
    else reject
  end
end
```

Figure 1. A non deterministic algorithm for a *co*NP-complete problem.

In conclusion we have that $L \in co\mathrm{NP}$ so that $\mathrm{NP} = co\mathrm{NP}$.

Conversely, let us assume that $\mathrm{NP} = co\mathrm{NP}$ and let L be the set of pairs (φ, τ) such that φ is a CNF-formula and τ is an optimum truth-assignment, that is, a truth-assignment satisfying the maximum number of clauses. Clearly, $L \in co\mathrm{NP}$ and thus $L \in \mathrm{NP}$. Let NT be a non-deterministic Turing machine deciding L in polynomial time and let A be the following NPO problem.

1. An instance is a CNF-formula φ.
2. For any CNF-formula φ, a feasible solution is a pair (τ, y) where τ is a truth-assignment for φ and y is a computation path of $NT(\varphi, \tau)$.
3. For any CNF-formula φ and for any feasible solution (τ, y),

$$m(\varphi, (\tau, y)) = \begin{cases} n/2 & \text{if } y \text{ is a rejecting computation,} \\ n & \text{otherwise} \end{cases}$$

where n denotes the length of φ.

Clearly, for any instance φ, $opt(\varphi) = n$ so that A admits a non-constructive polynomial-time approximation scheme. We now prove that A is APX-complete by considering the following PTAS-reduction from Max Sat to A.

1. For any CNF-formula φ and for any $r > 1$, $f(\varphi, r) = \varphi$.
2. For any CNF-formula φ, for any pair (τ, y) and for any $r > 1$,

$$g(\varphi, (\tau, y), r) = \begin{cases} \tau_{\mathrm{apx}} & \text{if } y \text{ is a rejecting computation,} \\ \tau & \text{otherwise} \end{cases}$$

where τ_{apx} denotes a 2-approximate solution of φ for Max Satisfiability (see [10]).
3. For any $r > 1$, $c(r) = r$.

In order to prove that the above reduction is indeed a PTAS-reduction it suffices to observe that, for any pair (τ, y), either $R_A(\varphi, (\tau, y)) = 1$ and τ is optimum for Max Satisfiability or $R_A(\varphi, (\tau, y)) = 2$ and the performance ratio of τ_{apx} is at most 2. □

The last result of this section shows that the PTAS-reducibility does not preserve the existence of non-constructive polynomial-time approximation scheme thus suggesting that, in order to investigate the relation between APX and ncPTAS, a 'non-constructive' version of the PTAS reducibility has to be introduced.

Theorem 5. *If* $\mathrm{P} \neq \mathrm{NP} \cap co\mathrm{NP}$*, then two approximable problems A and B exist such that A is* PTAS*-reducible to B, $B \in nc\mathrm{PTAS}$, but $A \notin nc\mathrm{PTAS}$.*

Proof. Let $L \in \mathrm{NP} \cap co\mathrm{NP} - \mathrm{P}$, let NT and NT^c be the non-deterministic Turing machines deciding L and L^c in polynomial time, respectively, and let q be a polynomial such that for any x an accepting computation of $NT(x)$ or an accepting computation of $NT^c(x)$ exists whose encoding is at most $q(|x|)$ bit long. We then define the NPO problem A as follows.

1. I contains any string x.
2. For any x, a feasible solution is any string y such that $|y| \leq q(|x|)$.
3. For any instance x of length n and for any solution y, $m_A(x, y) = 2n$ if y is the encoding of an accepting computation of $NT(x)$; $m_A(x, y) = n$ otherwise.

The NPO problem B is defined similarly to A, except that $m_B(x, y) = 2n$ if y is the encoding of an accepting computation of either $NT(x)$ or $NT^c(x)$, and $m_B(x, y) = n$ otherwise.

Note that (id, id, id) is a PTAS-reduction from A to B, where id is the identity function, because, for any string x and for any solution y of x, $R_A(x, y) \leq R_B(x, y)$. Indeed, either y is not the encoding of any accepting computation, and thus $R_A(x, y) \leq 2 = R_B(x, y)$, or y is the encoding of an accepting computation of $NT(x)$ or $NT^c(x)$, and thus $R_A(x, y) = R_B(x, y) = 1$.

It is also easy to see that $A, B \in \mathrm{APX}$ (since any solution is 2-approximate) and that $B \in \mathrm{ncPTAS}$ (since, for any x, $opt_B(x) = 2n$). Moreover $A \notin \mathrm{ncPTAS}$, because if T_A is a non-constructive polynomial-time approximation scheme, then $T_A(x, 3/2) = 2n$ if and only if $x \in L$, and this implies $L \in \mathrm{P}$. The theorem thus follows. □

4. Connections with the Parameterized Complexity

In order to obtain a deeper insight on the relative complexity of constructive and non-constructive approximation schemes, we have to define a generalization of the notion of associated language. The difficulty is that the value of an approximate solution is not exactly defined, since it is only expected to lay in a certain interval. For this reason, it is necessary to define a *class* of associated languages for every NPO problem.

Definition 13. *Let A be an NPO problem. A language L belongs to the class* $\mathrm{L}(A)$ *if and only if the following properties hold.*

1. *L contains triples (x, k, d) where x is an instance of A and k and d are positive integers.*
2. *If $(x, k, d) \in L$ then $opt(x) \geq k$.*
3. *If $(x, k, d) \notin L$ then $opt(x) < k(d + 1)/d$*

Let us fix any $d \geq 1$, let $r = (d+1)/d$ and let us consider any instance x and any integer $k \geq 0$. We can note that if $k > opt(x)$, then $(x, k, d) \notin L$, if $k \leq (1/r)opt(x)$ then $(x, k, d) \in L$, and if $(1/r)opt(x) < k \leq opt(x)$ then (x, k, d) may or may not belong to L.

Definition 14. *A language whose instances have the form (x, d), where d is a positive integer, belongs to the class* SP *if and only if an algorithm exists deciding the language in time $O(f(d)|x|^{g(d)})$, that is, in time polynomial in $|x|$ for every fixed d.*

The notion of fixed parameter complexity and of class SP is mainly due to Downey and Fellows (for an introduction to these ideas see [7]). The following theorem relates the fixed parameter complexity of the languages in $\mathrm{L}(A)$ to the complexity of evaluating approximate values.

Theorem 6. *An NPO problem A belongs to nc*PTAS *if and only if* $\mathrm{L}(A) \cap \mathrm{SP} \neq \emptyset$.

Proof. Let A be an NPO problem in ncPTAS, and let T be the non-constructive polynomial-time approximation scheme for A. We define a language L^T as follows: $(x, k, d) \in L^T$ if and only if $T(x, 1+1/d) \geq k$ (note that $L \in \mathrm{SP}$). If $(x, k, d) \in L^T$, then $k \leq T(x, 1+1/d) \leq opt(x)$, while if $(x, k, d) \notin L^T$, then $opt(x) \leq T(x, 1+1/d)(d+1)/d < k(d+1)/d$. Thus, $L^T \in \mathrm{L}(A)$.

Conversely, let $L \in \mathrm{L}(A) \cap \mathrm{SP}$. For any fixed $r > 1$, the algorithm of Fig. 2 finds a value k such that $(1/r)opt(x) \leq k \leq opt(x)$.

```
begin
  $d := \lceil 3/(r-1) \rceil$;
  $m := 1$;
  while $(x, m, d) \in L$ do
    $m := (1 + 1/d)m$;
  return $\lceil md/(d+1) \rceil$
end
```

Figure 2. A non-constructive polynomial-time approximation scheme

The algorithm returns a value $k = \lceil md/(d+1) \rceil$ where m is such that $(x, m, d) \notin L$, that is, $m > opt(x)d/(d+1)$. Since $(x, md/(d+1), d) \in L$ then $opt(x) \geq md/(d+1)$. By the definition of NPO problem, it follows that $opt(x)$ is a positive integer, so that

$$opt(x)(d/(d+1))^2 < md/(d+1) \leq opt(x)$$

implies

$$opt(x)(d/(d+1))^2 \leq k \leq opt(x) \ .$$

Moreover, $((d+1)/d)^2 \leq 1 + 3/d \leq r$, and thus $(1/r)opt(x) \leq k \leq opt(x)$. The running time of the algorithm is polynomial in the length of x for any fixed $r > 1$. □

5. Conclusions

In this paper, we have shown the first example of a polynomially bounded APX-complete problem. To this aim, we have introduced a new approximation scheme preserving reducibility. We have also presented several results regarding the relation between constructive and non-constructive approximation of NPO problems.

With respect to these latter results, we believe that similar relations should hold in the case of the classes APX and ncAPX, even though a different notion of reducibility has to be defined (observe that the PTAS-reducibility does not preserve membership in APX). The AP-reducibility introduced in [4] could be one reasonable possibility.

Finally, we conjecture that stronger relations hold between the theory of approximation of NPO problems and the fixed parameter complexity theory. In [3] some other results are presented along this line of research.

References

1. S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and hardness of approximation problems. In *Proceedings of the 33rd IEEE Symposium on Foundations of Computer Science*, pages 14–23, 1992.
2. G. Ausiello, P. Crescenzi, and M. Protasi. Approximate solution of NP optimization problems. Technical Report SI/RR-94/03, Dip. di Scienze dell'Informazione, Università di Roma "La Sapienza", March 1994. (to appear in *Theoretical Computer Science*).
3. L. Cai and J. Chen. On fixed-parameter tractability and approximability of NP optimization problems. In *Proceedings of 2nd Israel Symposium on Theory of Computing Systems*, pages 118–126, 1993.
4. P. Crescenzi, V. Kann, R. Silvestri, and L. Trevisan. Structure in approximation classes. Technical Report SI/RR-95/07, Dip. di Scienze dell'Informazione, Università di Roma "La Sapienza", April 1995.
5. P. Crescenzi and A. Panconesi. Completeness in approximation classes. *Information and Computation*, 93:241–262, 1991.
6. P. Crescenzi and R. Silvestri. Relative complexity of evaluating the optimum cost and constructing the optimum for maximization problems. *Information Processing Letters*, 33:221–226, 1990.
7. R.G. Downey and M.R. Fellows. Fixed-parameter intractability. In *Proceedings of the 7th IEEE Conference on Structure in Complexity Theory*, pages 36–49, 1992.
8. L.A. Hemaspaandra. Complexity theory column 5: the not-ready-for-prime-time conjectures. *SIGACT News*, 1994.
9. O. H. Ibarra and C. E. Kim. Fast approximation for the knapsack and sum of subset problems. *Journal of the ACM*, 22:463–468, 1975.
10. D.S. Johnson. Approximation algorithms for combinatorial problems. *Journal of Computer and System Sciences*, 9:256–278, 1974.
11. V. Kann. *On the Approximability of NP-Complete Optimization Problems.* PhD thesis, Department of Numerical Analysis and Computing Science, Royal Institute of Technology, Stockolm, 1992.
12. S. Khanna, R. Motwani, M. Sudan, and U. Vazirani. On syntactic versus computational views of approximability. In *Proceedings of the 35th IEEE Symposium on Foundations of Computer Science*, 1994.
13. M.W. Krentel. The complexity of optimization problems. *Journal of Computer and System Sciences*, 36:490–509, 1988.
14. P. Orponen and H. Mannila. On approximation preserving reductions: complete problems and robust measures. Technical Report C-1987-28, Department of Computer Science, University of Helsinki, 1987.
15. C. H. Papadimitriou and M. Yannakakis. Optimization, approximation, and complexity classes. *Journal of Computer and System Sciences*, 43:425–440, 1991.

16. A. Paz and S. Moran. Non deterministic polynomial optimization problems and their approximation. *Theoretical Computer Science*, 15:251–277, 1981.

17. L.G. Valiant. Relative complexity of cheking and evaluating. *Information Processing Letters*, 5(1):20–23, 1976.

18. K. Wagner. Bounded query computations. In *Proceedings of the 3th IEEE Conference on Structure in Complexity Theory*, pages 260–277, 1988.