On Approximation Scheme Preserving Reducibility and Its Applications*

Extended abstract

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.

Abstract. In this paper we generalize the notion of polynomial-time approximation scheme preserving reducibility, called PTAS-reducibility, introduced in [4]. 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 in [10] 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 ϵ-approximate value and computing an ϵ-approximate solution for any ϵ. 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 [8].

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 ϵ for a given ϵ and the class of problems that are approximable within any factor ϵ. It is well-known that PTAS is strictly contained in APX if and only if $P \neq NP$.

* Research partially supported by the MURST project *Algoritmi, Modelli di Calcolo, Strutture Informative.*

Several notions of approximation scheme preserving reducibilities have been introduced in [4, 11, 12] with the aim of establishing hardness and completeness results in APX and of deriving proofs of intractability of arbitrary approximation from them (see also Chap. 3 of [9] for a survey on the notion of reducibility among optimization problems). In particular, in [4] 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 [12] 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 [10] 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 ϵ-approximate value and computing an ϵ-approximate solution for any ϵ. The relative complexity of checking and evaluating a function was first considered in [14]. 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 P= NP. In [13] and, successively, in [5] the relative complexity of evaluating the optimum cost 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 P$\neq$ NP $\cap$ *co*NP then a problem exists in APX $-$ PTAS whose optimum cost 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 NP = *co*NP.

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. Finally, further results concerning the notion of non-constructive approximation preserving reducibility and of APX-intermediate problem are presented.

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. The set $sol(x)$ is recognizable in polynomial time and a polynomial p exists such that, for any $y \in sol(x)$, $|y| \leq p(|x|)$.
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

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

In the following sol^* will denote the multi-valued function mapping an instance x to the set of optimum solutions, while m^* 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

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

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

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

Definition 4. An NPO problem A belongs to the *class* APX if a rational $\epsilon \in (0, 1)$ and a polynomial-time algorithm T exist such that, for any instance x, the performance ratio of the feasible solution $T(x)$ is at least ϵ.

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 $\epsilon \in (0,1)$, $T(x,\epsilon)$ returns a feasible solution whose performance ratio is at least ϵ in time bounded by $q(|x|)$ where q 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 ncPTAS* 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 $\epsilon \in (0,1)$, $T(x,\epsilon)$ returns a value between $\epsilon m^*(x)$ and $m^*(x)$ in time bounded by $q(|x|)$ where q is a polynomial.

Observe that the time complexity of a (non-constructive) approximation scheme in the last two definitions may be exponential in the rational $1/(1-\epsilon)$, that is, it may be of the type $2^{1/(1-\epsilon)}p(|x|)$ or $|x|^{1/(1-\epsilon)}$ 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 $\epsilon \in (0,1)$, $f(x,\epsilon) \in I_B$ is computable in time polynomial with respect to $|x|$.
2. For any $x \in I_A$, for any $\epsilon \in (0,1)$, and for any $y \in sol_B(f(x,\epsilon))$, $g(x,y,\epsilon) \in sol_A(x)$ is computable in time polynomial with respect to both $|x|$ and $|y|$.
3. $c : (0,1) \rightarrow (0,1)$.
4. For any $x \in I_A$, for any $\epsilon \in (0,1)$, and for any $y \in sol_B(f(x,\epsilon))$,

$$R_B(f(x,\epsilon),y) \geq c(\epsilon) \text{ implies } R_A(x,g(x,y,\epsilon)) \geq \epsilon \ .$$

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

Remark. The PTAS-reducibility is a generalization of a reducibility introduced in [4] 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 ϵ. Moreover, in [12] 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 $c(\epsilon) = 1 - \frac{1-\epsilon}{\alpha\beta}$ where α and β are constants.

Proposition 8. *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

$$T_A(x, \epsilon) = g(x, T_B(f(x, \epsilon), c(\epsilon)), \epsilon)$$

is a polynomial-time approximation scheme for A. □

Definition 9. An NPO problem A in APX is APX-complete if, for any other problem B in APX, $B \leq A$.

In [11] 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 [4].

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
$$W \leq \sum_{i=1}^{n} w_i \leq 2W \quad ,$$
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, which 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 n denotes the number of variables and m_{MBWS} and m_{MPBWS} denote the measure functions of MBWS and MPBWS, respectively (this 'scaled' version of MPBWS was first suggested in [10]). 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 10. 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 ϵ).

1. For any $\epsilon \in (0,1)$, $f(x,\epsilon) = x$.
2. For any τ and for any $\epsilon \in (0,1)$,

$$g(x,\tau,\epsilon) = \begin{cases} \tau & \text{if } \epsilon < (n-1)/n, \\ \tau^* & \text{otherwise} \end{cases}$$

   where τ^* denotes an optimum solution for MBWS.
3. For any $\epsilon \in (0,1)$, $c(\epsilon) = (\epsilon + 1)/2$.

Observe that, according to the definition of the PTAS-reducibility, the running time of g can be exponential in $1/(1-\epsilon)$. If $\epsilon \geq (n-1)/n$ 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 $\epsilon < (n-1)/n$ and that τ is any truth assignment. Let

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

Then

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

while

$$\begin{aligned} R_{\text{MBWS}}(x,\tau) &= \frac{m_{\text{MBWS}}(x,\tau)}{m^*_{\text{MBWS}}(x)} \\ &\geq \frac{m^*_{\text{MBWS}}(x) - (i_{\tau^*} - i_\tau + 1)W/n}{m^*_{\text{MBWS}}(x)} \\ &= 1 - \frac{(i_{\tau^*} - i_\tau + 1)W/n}{m^*_{\text{MBWS}}(x)} \\ &\geq 1 - \frac{(i_{\tau^*} - i_\tau + 1)W/n}{W(1 + i_{\tau^*}/n)} \\ &= 1 - \frac{i_{\tau^*} - i_\tau + 1}{n + i_{\tau^*}}. \end{aligned}$$

If $i_{\tau^*} - i_\tau = 0$ then

$$R_{\text{MBWS}}(x,\tau) \geq \frac{n + i_{\tau^*} - 1}{n + i_{\tau^*}} \geq \frac{n-1}{n} > \epsilon .$$

Otherwise

$$R_{\mathrm{MBWS}}(x,\tau) \geq 1 - \frac{i_{\tau^*} - i_\tau + 1}{n + i_{\tau^*}} \geq 1 - 2\frac{i_{\tau^*} - i_\tau}{n + i_{\tau^*}} = 2R_{\mathrm{MPBWS}}(x,\tau) - 1 \ .$$

If $R_{\mathrm{MPBWS}}(x,\tau) \geq c(\epsilon)$, then $R_{\mathrm{MBWS}}(x,\tau) \geq 2c(\epsilon) - 1 = \epsilon$.

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

Corollary 11. 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 [10] 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. In the next section we shall see that they do not admit a non-constructive polynomial-time approximation scheme either unless NP = *co*NP.

Remark. By making use of either the P-reducibility or the L-reducibility 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 both these two reducibilities do not allow the function g to depend on ϵ: as a consequence, this function is forced to map optimum solutions into optimum solutions. More formally, it is possible to prove that the APX-completeness of a polynomially bounded optimization problem with respect to either the P-reducibility or the L-reducibility would imply that $\mathrm{P}^{\mathrm{SAT}} = \mathrm{P}^{\mathrm{SAT}[\log n]}$ where $\mathrm{P}^{\mathrm{SAT}}$ (respectively, $\mathrm{P}^{\mathrm{SAT}[\log n]}$) denotes the class of languages decidable in polynomial time asking a polynomial (respectively, logarithmic) number of queries to an oracle for the satisfiability problem. This latter event seems to be unlikely and, in any case, the relationship between these two classes is a well-known open question in complexity theory [7].

3 Evaluating, Constructing, and APX-Completeness

Assume that P $\neq$ NP $\cap$ *co*NP. Let $L \in$ NP $\cap$ *co*NP $-$ 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 maximization 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 $A \in nc$PTAS. Indeed, the following algorithm is a non-constructive polynomial-time approximation scheme: for any instance (x, φ) and for any ϵ,

$$T((x,\varphi),\epsilon) = \begin{cases} n & \text{if } \epsilon \leq (n-1)/n \text{ or} \\ & \epsilon > (n-1)/n \text{ and } \varphi \text{ is not satisfiable,} \\ n+1 & \text{if } \epsilon > (n-1)/n \text{ and } \varphi \text{ is satisfiable} \end{cases}$$

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 $(y, \tau) = T((x, \varphi), 2/3)$. 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 the following result.

Theorem 12. *If* P$\neq$ NP $\cap$ *co*NP *then an* NP*-hard maximization problem exist that belongs to* APX $\cap$ *nc*PTAS $-$ 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 ϵ-approximate values and computing ϵ-approximate solutions for any ϵ are computationally hard.

To this aim, let us firts recall a result obtained in [1].

Theorem 13 [1]. *Let L be a language in NP. Then a rational $\epsilon \in (0,1)$ 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 ϵ of all the clauses of φ_x.*

As a consequence of this result, it follows that Max Sat, that is, the problem of maximizing the number of satisfied clauses in a given CNF-formula, does not admit a polynomial-time approximation scheme unless P= NP (note that it is well-known that this problem is approximable [8, 15]).

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 13, it follows that, for any x, the CNF-formula φ_x satisfies the following two implications: (a) if $x \in L$ then $m^*_{\mathrm{MSAT}}(\varphi_x) = m$ and (b) if $x \notin L$ then $m^*_{\mathrm{MSAT}}(\varphi_x) < \epsilon m$ 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 $f(\varphi_x, \epsilon)$ whose performance ratio is at least $c(\epsilon)$, then $\tau_y = g(\varphi_x, y, \epsilon)$ is truth assignment whose performance ratio is at least ϵ.

It is then easy to verify that τ_y satisfies less than ϵm clauses of φ_x if and only if $x \notin L$. Indeed, if $x \notin L$ then any truth assignment satisfies less than ϵm clauses. Conversely, if τ_y satisfies less than ϵm clauses then $m^*_{\mathrm{MSAT}}(\varphi_x) \leq m_{\mathrm{MSAT}}(\varphi_x, \tau_y)1/\epsilon < m$, 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 performs, for any x, the following steps.

1. Compute φ_x and set m the number of its clauses.
2. Compute $a = T(f(\varphi_x, \epsilon), c(\epsilon))$.
3. Guess a feasible solution y of $f(\varphi_x, \epsilon)$: if its measure is less than a then reject. Otherwise compute $\tau_y = g(\varphi_x, y, \epsilon)$: accept if and only if τ_y satisfies less than ϵm clauses of φ_x.

In conclusion we have proved the following result.

Theorem 14. *If A is an* APX*-complete problem that admits a non-constructive polynomial-time approximation scheme then* NP = *co*NP.

A natural question that arises from the above theorem is whether a stronger evidence can be given for the non-existence of non-constructive approximation schemes for APX-complete problems. Unfortunately, the following result shows that this is not possible.

Theorem 15. *If* NP = *co*NP *then an* APX*-complete problem exists that admits a non-constructive polynomial-time approximation scheme.*

Proof. 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, $L \in co$NP and thus $L \in$ NP. Let NT be a non-deterministic Turing machine deciding L in polynomial time and let A be the following maximization 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, $m^*(\varphi) = n$ 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 $\epsilon \in (0, 1)$, $f(\varphi, \epsilon) = \varphi$.
2. For any CNF-formula φ, for any pair (τ, y) and for any $\epsilon \in (0, 1)$,

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

where τ_{apx} denotes a 1/2-approximate solution of φ for MAX SAT (see [8]).
3. For any $\epsilon \in (0, 1)$, $c(\epsilon) = \epsilon$.

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 SAT or $R_A(\varphi, (\tau, y)) \leq 1/2$ and the performance ratio of τ_{apx} is at least 1/2. □

4 Further Results

4.1 Relationships with 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 maximization problem.

Definition 16. Let A be a maximization problem. A language L belongs to the class 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 $m^*(x) > k(1 - 1/d)$.
3. If $(x, k, d) \notin L$ then $m^*(x) \leq k$

Definition 17. 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 [6]). The following theorem relates the fixed parameter complexity of the languages in L(A) to the complexity of evaluating approximate values.

Theorem 18. *A maximization problem A admits an ncPTAS if and only if* $\mathrm{L}(A) \cap \mathrm{SP} \neq \emptyset$.

4.2 Non-Constructive Reducibilities

The notion of non-constructiveness may also be applied to the study of reducibilities among optimizations problems. The ncPTAS-reducibility can be defined similarly to the PTAS-reducibility. The main property of this reducibility is that if A is ncPTAS-reducible to B and $B \in nc$PTAS, then $A \in nc$PTAS.

It is worth noting that if a problem A is APX-complete with respect to the ncPTAS-reducibility, then it is NP-hard to evaluate approximate values for A. Thus constructing approximate solutions for A is reducible to evaluating approximate values.

Oddly enough, the nice property of preserving non-constructive approximation schemes is not achieved by PTAS-reducibility.

Theorem 19. *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 but A is not nc*PTAS-*reducible to B.*

The following result holds as well.

Theorem 20. *If* $\mathrm{P} \neq \mathrm{NP} \cap co\mathrm{NP}$*, then two approximable problems A and B exist such that A is nc*PTAS-*reducible to B but A is not* PTAS-*reducible to B.*

4.3 APX-intermediate Problems

An NPO problem A in APX − PTAS is APX-intermediate if it is not APX-complete. An interesting consequence of Theorem 14 is that if a certain approximable problem belongs to ncPTAS − PTAS, then it is APX-intermediate unless NP = coNP. This observation allows us to show that the existence of a 'natural' APX-intermediate problem is strictly related to the existence of many-to-one one-way functions [5] (thus partially solving an open question raised in [4]). More recently, however, it has been shown in [3] that the Bin Packing problem is APX-intermediate unless the polynomial-time hierarchy collapses.

References

1. Arora S., Lund C., Motwani R., Sudan M., Szegedy M.: Proof verification and hardness of approximation problems. Proc. 33-rd Ann. IEEE Symp. on Foundations of Computer Science (1992) 14–23

2. Ausiello G., Crescenzi P., Protasi M.: Approximate Solution of NP Optimization Problems Technical Report SI/RR-94/03, Università di Roma "La Sapienza" (1994)
3. Crescenzi P., Kann V., Trevisan L.: On the structure of the classes NPO and APX. Manuscript in preparation (1994)
4. Crescenzi P., Panconesi A.: Completeness in approximation classes. Information and Computation **93** (1991) 241–262
5. Crescenzi P., Silvestri R.: Relative complexity of evaluating the optimum cost and constructing the optimum for maximization problems. Information Processing Letters **33** (1990) 221–226
6. Downey R.G., Fellows M.R.: Fixed parameter intractability. Proc. 7-th Structure in Complexity Theory (1992) 36–49
7. Hemaspaandra, L.A.: Complexity theory column 5: the not-ready-for-prime-time conjectures. SIGACT News (1994)
8. Johnson D.S.: Approximation algorithms for combinatorial problems. J. Comput. and Syst. Sci.**9** (1974) 256–278
9. Kann V.: On the approximability of NP-complete optimization problems. PhD thesis, Department of Numerical Analysis and Computing Science, Royal Institute of technology, Stockholm (1992)
10. Khanna S., Motwani R., Sudan M., Vazirani U.: On syntactic versus computational views of approximability. Proc. 35-th Ann. IEEE Symp. on Foundations of Computer Science (1994) to appear
11. Orponen P., Mannila H.: On approximation preserving reductions: complete problems and robust measures. Technical Report C-1987-28, Department of Computer Science, University of Helsinki (1987)
12. Papadimitriou C.H., Yannakakis M.: Optimization, approximation, and complexity classes. J. Comput. and Syst. Sci. **43** (1991), 425–440
13. Paz A., Moran S.: Non deterministic polynomial optimization problems and their approximations. Theoret. Comput. Sci. **15** (1981) 251–277
14. Valiant L.: Relative complexity of checking and evaluating. Information Processing Letters **5** (1976) 20–23
15. Yannakakis M.: On the approximation of maximum satisfiability. Proc. 3-rd ACM-SIAM Symp. on Discrete Algorithms (1992) 1–9

This article was processed using the LaTeX macro package with LLNCS style