Theoretical Computer Science 12 (1980) 83–96
© North-Holland Publishing Company

TOWARD A UNIFIED APPROACH FOR THE CLASSIFICATION OF NP-COMPLETE OPTIMIZATION PROBLEMS*

G. AUSIELLO and A. MARCHETTI-SPACCAMELA
Istituto di Automatica, Consiglio Nazionale Delle Ricerche CSSCCA, 00184 Roma, Italy

M. PROTASI**
Universita' di Lecce, Istituto Matematico, Lecce, Italy

Communicated by E. Engeler
Received March 1979
Revised August 1979

Abstract. Two notions which have been introduced with the aim of classifying NP-complete optimization problems are compared: the notion of strong NP-completeness, due to Garey and Johnson, and that of simple and rigid problems, due to Paz and Moran. In particular, we show under what conditions reductions preserve rigidity, simplicity, strong simplicity and p-simplicity and we show that under reasonable hypotheses, p-simple problems are solved by pseudo-polynomial algorithms and strong NP-complete problems are weakly rigid.

1. Introduction

NP-complete optimization (NPCO) problems play a major role in discrete mathematics and operations research due to their relevance in many practical problems and the consequent need to study their properties and to find efficient, exact or well approximate algorithms for them.

Also in theoretical computer science NP-complete optimization problems have become an interesting research area since when results by Garey and Johnson, Sahni and others (see, for example, [3, 12]) have shown differences among these problems with respect to their combinatorial structure and their properties of approximability and have, hence, suggested the need to classify NPCO problems.

Here we compare two approaches to the classification of NPCO problems: the approach of Paz and Moran [10, 11] and the approach of Garey and Johnson [4].

Paz and Moran introduce a classification of NPCO problems based on the fact that considering only those inputs of the problem whose optimal value is bounded by an integer, it is possible to divide all the problems in different classes as regards their computational complexity (rigid, simple and p-simple problems). Furthermore these classes are then related to the approximability properties of the problem.

* A first version of this work was presented at Frege Conference, Jena, 1979.
** Present address: Università dell'Aquilo, Istituto Matematico, L'Aquila, Italy

Under a different approach Garey and Johnson give another characterization which is based on the concept of strong NP-complete problem (a problem is strong NP-complete when limiting ourselves to those inputs, whose 'value' is bounded by a polynomial in the length of the input, we still obtain an NP-complete problem) and of pseudopolynomial algorithm (an algorithm which is polynomial in the length of the input and in the magnitude of the greatest number occurring in the instance). Also in this case interesting relations among these concepts and approximation properties are stated.

These papers, without any doubt, capture important concepts. Nevertheless, it seems that an attempt of organizing all these results in a unified framework as general as possible is lacking. Furthermore any effort of comparison among different approaches has not been sufficiently developed.

The aim of our paper is therefore to make a first step in this direction. Starting from the observation that, intuitively, there is a similarity among some of the consequences of Paz and Moran, Garey and Johnson approaches, we have introduced a formal framework in which it is possible to establish clear connections among different concepts of the two approaches, at least under restricted but reasonable hypotheses. So, we have established under what conditions a p-simple problem is pseudo-polynomial and a strong NP-complete problem is weakly rigid. Besides, our point of view allows to derive some new consequences both concerning the classification of problems and the characterization of reductions that exist among different problems. We have stated what conditions must be satisfied to have a polynomial reduction from a rigid problem to a simple problem, from a p-simple problem to a p-simple problem and so on. Finally it seems interesting to us that some of these results can be interpreted as a formalization of facts that are used in practice when studying the solution of a particular problem, such as, for example, the fact that a problem with polynomially bounded objective function cannot be fully approximated.

In particular, in Section 2 we give the basic terminology and notation. In Section 3 we very briefly summarize the Paz and Moran approach with a slightly different formulation, giving new results such as those above stated concerning the characterization of reductions among problems belonging to different classes. In Section 4 after recalling the main definitions and results of Garey and Johnson approach, we establish under what conditions the results of these two approaches can be compared, eventually exhibiting some examples which show that violating the conditions, the two approaches lead to different conclusions in the classification of NP-complete problems.

2. Basic concepts and terminology

In order to establish a formal ground for the study of the properties of optimization problems we first give an abstract notion of optimization problem which is broad enough to include most common problems of this kind. Following the literature [7], we consider an *NP-optimization problem* to be characterized by a polynomially decidable set INPUT of instances, a polynomially decidable set OUTPUT of possible outcomes, a mapping SOL:INPUT$\to P$(OUTPUT) which, given any instance x of the problem, in polynomial time nondeterministically provides the *feasible solutions* of x and a mapping m:OUTPUT$\to Z$ (where Z is the set of all integers) which in polynomial time provides the *measure* of a feasible solution (if A is a maximization problem) or its additive inverse (if A is a minimization problem).

We will denote $\tilde{m}(x)$ and $m^*(x)$ the worst and (respectively) the best solution of x with respect to the ordering of Z. In this paper we assume that the worst solution can be easily (in polynomial time) determined. This is what happens in most interesting cases in which the worst solution actually is a trivial solution.

Since we are interested in studying those optimization problems which are 'associated' to NP-complete recognition problems we restrict ourselves to considering a particular class of NP-complete problems:

Definition 1. Let A be an NP optimization problem. The *combinatorial problem associated* to A is the set

$$A^c = \{\langle x, k\rangle \mid x \in \mathrm{INPUT}_A \text{ and } k \leq m^*(x) - \tilde{m}(x)\}.$$

On the base of this definition we exclude from our study those problems which are not directly related to optimization problems.[1]

If A^c is NP-complete we say A is an NP-complete optimization (NPCO) problem.

Example. The problem MAX–CLIQUE is an NPCO problem. It is characterized by
- INPUT = set of all finite graphs,
- OUTPUT = set of all finite complete graphs,
- SOL(x) = set of all complete subgraphs of a graph x,
- $m(y)$ = number of nodes of y.

The associated combinatorial problem $\{\langle x, k\rangle \mid x$ has a complete subgraph of at least k nodes$\}$ is a well-known NP-complete recognition problem. In this case $\tilde{m}(x) = 1$ is clearly the trivial solution of the optimization problem.

A more adequate and broad presentation of NPCO problems with general motivations and examples is given in [1].

For this particular class of NP-complete recognition problems the concept of reduction [8] can be specialized and it can be extended to the associated optimization problems.

[1] In their paper Paz and Moran [11] suggest that any NP recognition problem can be represented as an optimization problem but we prefer a more straightforward and explicit definition.

Definition 2. Let A and B be two NPCO problems. We say that A is polynomially reducible to $B(A: \leqslant_p B)$ if there exist two polynomially computable functions

$$f_1 : \mathrm{INPUT}_A \to \mathrm{INPUT}_B, \qquad f_2 : \mathrm{INPUT}_A \times Z \to Z$$

such that

$$\langle f_1(x), f_2(x, k)\rangle \in B^c \quad \text{iff} \quad \langle x, k\rangle \in A^c.$$

Throughout this paper we will deal only with this kind of reductions. For simplicity we will say A reducible to B and we will drop the subscript p from $\leqslant_p$.

Since we are interested in discussing the approximability of NPCO problems and reductions between problems with a different behaviour with respect to this property, we first give some basic definitions that introduce the concept of approximate algorithm, of approximable problem and of fully approximable problem [11, 12].

Definition 3. Let A be an NPCO problem. We say that

(i) A_p is an *approximate algorithm* for A if given any $x \in \mathrm{INPUT}_A$ $A_p(x)$ is in $\mathrm{SOL}(x)$ and A_p is computable in polynomial time;

(ii) A_p is an *ε-approximate algorithm* for A if it is an approximate algorithm for A and for every $x \in \mathrm{INPUT}_A$[2]

$$F_{A_p}(x) = \left|\frac{m^*(x) - m(A_p(x))}{m^*(x) - \tilde{m}(x)}\right| \leqslant \varepsilon.$$

Definition 4. Let A be an NPCO problem; we say that

(i) A is *approximable* if given any $\varepsilon > 0$ there exists an ε-approximable algorithm;

(ii) A is *fully approximable* if there exists a polynomial $\lambda x \lambda y[q(x, y)]$ such that for every ε there exists an ε-approximate algorithm A_p that runs in time bounded by $q(|x|, 1/\varepsilon)$.

Many results in the recent literature are devoted to establishing whether a given problem is approximable or fully approximable or it cannot be approximated. For example it is known that the MAX–SUBSET–SUM problem is fully approximable while MAX–CLIQUE is not fully approximable. A list of papers dealing with results in this area is provided by Garey and Johnson [3]. At present no natural characterization of the class of problems which are approximable or fully approximable is known. The results given by Paz and Moran [11] and Garey and Johnson [4] are nevertheless an important step forward in this direction. For this reason our aim has

[2] Note that we prefer to use an evaluation function different from the standard one, because we want to deal with maximization and minimization problems in a symmetrical way. On the other side, in [13, 14] a study of evaluation functions is done, showing that some natural properties are not satisfied by the classical evaluation function. However, we stress that the results given in this paper still hold using the old evaluation function, provided the measure of the trivial solution is always assumed to be equal to zero.

been to determine conditions for the comparison of these results and at the same time to develop this kind of research and to derive consequences which are useful for a better understanding of the properties of NP-complete optimization problems.

3. Truncated combinatorial problems and their properties

The first approach [11] to the characterization of NP-complete optimization problems is based on the complexity of the recognition of an infinite sequence of bounded subsets of the associated combinatorial problem.

Informally, if we consider the search space that has to be explored in order to find feasible solutions to an NP-optimization problem we may observe the following facts:

(i) if the size of the search space is polynomially related to the size of the input, the problem itself is polynomially solvable;

(ii) otherwise an a priori evaluation of the size of the search space shows that it grows exponentially.

Let us suppose now to look for solutions whose measure does not exceed a certain bound k; in many cases the size of the search space is polynomially related to the size of the input.

A typical example of this kind of problems is the problem MAX–CLIQUE in which the complete subgraphs of size k in a graph of size n are at most $\binom{n}{k}$, that is their number is polynomial in n. Since this does not happen in all cases it suggests the following definition.

Definition 5. Let A be a NPCO problem; let A^c be the associated combinatorial problem. A *truncated combinatorial problem* of A is a set

$$A_{\boldsymbol{k}}^c = \{\langle x, k\rangle \in A^c \mid k \leq \boldsymbol{k}\},$$

where $\boldsymbol{k}$ is any nonnegative integer ($\boldsymbol{k} \in N \cup \{0\} = N^*$).

Note that the sequence $\{A_{\boldsymbol{k}}^c\}_{\boldsymbol{k}=0}^{\infty}$ approximates the set A^c in a sense which is analogous to the definition of limit recursion approximation [5].

Definition 6. A is *simple* if, for every $\boldsymbol{k}$, $A_{\boldsymbol{k}}^c$ is polynomially decidable. A is *rigid* if it is not simple.

Note that if A is rigid there exists an integer $\boldsymbol{k}$ such that $A_{\boldsymbol{k}}^c$ is P-complete that is $A_{\boldsymbol{k}}^c$ is in P if and only if P = NP. If P = NP no rigid problem can exist. Examples of simple NPCO problems, besides MAX–CLIQUE, are MAX–SATISFIABILITY, MIN–CHROMATIC NUMBER.

Definition 5 and 6 are slightly modified with respect to the corresponding definitions in [11]. In fact we always start from the set A_0^c in which all pairs $\langle x, \tilde{m}(x)\rangle$ are included and, as long as $\boldsymbol{k}$ increases, we go further and further from the worst solution to the optimal solution. Due to this modification the class of simple problems is larger than the one defined by Paz and Moran [11].

For example the problem MIN–CHROMATIC–NUMBER which is rigid according to the original definitions (see [11]), is simple in our case and this is because, given any h, the set of possible colourings of a graph of N nodes with N–h colours has size which is polynomial in N.

On the other side, also in our terminology, the distinction between simple and rigid problems remains meaningful. Indeed MAX–WEIGHTED–SATISFIABILITY is rigid also according to our definitions. The definition of this problem and the proof of its rigidity is given by Ausiello et al. [2].

Note that if a problem is simple, then its worst solution is actually a trivial solution, that is it can be always found in polynomial time.

The concept of simple problem can be strengthened in the following way:

Definition 7. An NPCO problem A is *p-simple* if there is a polynomial Q such that, for every k, A_k^c is recognizable in determistic time bounded by $Q(|x|, k)$.

Typical examples of p-simple problems are MAX–SUBSET–SUM, JOB–SEQUENCING–WITH–DEADLINES etc., while the above listed simple problems are not p-simple.

Beside offering a first classification of NPCO problems, the concepts of simplicity and p-simplicity are relevant because it has been proven by Paz and Moran [11] that a necessary condition for a problem A to be approximable (fully-approximable) is that A is a simple (p-simple) NPCO problem and clearly these properties still hold under our definitions. So if we want to prove that a problem is not fully approximable it is sufficient to show that it is not p-simple, but in order to prove that a problem is not p-simple it is very hard to show that no algorithm which is polynomial in $|x|$ and k can exist. Much easier is to use the following definitions.

Definition 8. An NPCO problem A is *strongly simple* if, given any polynomial q, $A_q^c = \{\langle x, k\rangle \in A^c \mid k \leq q(|x|)\}$ is decidable in polynomial time. A is *weakly rigid* if there exists a polynomial p such that A_p^c is NP-complete.

Since a p-simple problem is strongly simple, to show that a problem is weakly rigid is a very easy method to prove that a problem is not p-simple and therefore not fully approximable. For example weakly rigid problems are MAX–CLIQUE, MAX–SATISFIABILITY, MIN–CHROMATIC–NUMBER and the proof is based on the fact that, for all these problems, for $q(n)$ increasing more rapidly than n, $A_q^c = A^c$.

This fact suggests an even easier condition that is sufficient for a problem not to be fully approximable.

Proposition 1. *Let A be an* NPCO *problem. If there exists a polynomial p such that, for all $x \in \mathrm{INPUT}_A$, $m^*(x) - \tilde{m}(x) \leq p(|x|)$, then A is not fully approximable.*

Proof. In fact in order to be fully approximable, A should satisfy the property that A_p^c is recognizable in polynomial time but, by hypothesis, we have that $A_p^c = A^c$ and, hence, A_p^c is NP-complete.

For some problems, like MAX–CLIQUE and MIN–CHROMATIC–NUMBER, Proposition 1 can be immediately applied. In fact in these cases $p(|x|) = |x|$.

In some other case, in order to apply Proposition 1, we may prove a stronger result that is useful for showing that a problem is weakly rigid.

Theorem 1. *Let A and B be two* NPCO *problems; if there exists a reduction $f = \langle f_1, f_2\rangle$ from A to B such that f satisfies the following property: $f_2(x, k) \leq p(|f_1(x)|)$ for some polynomial p and all $x \in \mathrm{INPUT}_A$, $k \in N^*$, then B is not fully approximable.*

Proof. If B was fully approximable, then for every polynomial q we should have B_q^c recognizable in polynomial time.

If we could recognize within polynomial time B_p^c, then we could also recognize A^c in polynomial time.

In fact, given a pair $\langle x, k\rangle$, we could compute in polynomial time $f_1(x)$ and $f_2(x, k)$ and since $f_2(x, k) \leq p(|f_1(x)|)$ we could use the decision procedure for B_p^c to check whether $\langle f_1(x), f_2(x, k)\rangle \in B_p^c$.

Note that in Theorem 1 the condition on f_2 may regard only a subset of B while in Proposition 1 all inputs must satisfy the hypothesis that $(m^*(x) - \tilde{m}(x)) \leq p(|x|)$.

Furthermore Theorem 1 partially characterizes the reductions between an arbitrary problem and a weakly rigid one. For example if we consider the trivial reduction (inclusion) from SIMPLE–MAX–CUT to MAX–CUT, we see that the image of SIMPLE–MAX–CUT is a subset of MAX–CUT where the measure is bounded by the number of nodes of the graph and this fact is sufficient to deduct that MAX–CUT is not fully approximable.

In the following we will continue the study of the characterization of reductions between problems belonging to different classes, and we will show how some of the considered properties can be inherited by polynomial reduction, under some natural hypothesis.

Theorem 2. *Let A and B be two* NPCO *problems such that $A \leq B$ via the reduction $f = \langle f_1, f_2\rangle$; if A is rigid and if there exists a monotone function g such that, for every $x \in \mathrm{INPUT}_A$, $k \in N^*$, $f_2(x, k) \leq g(k)$, then B is rigid.*

Proof. If A is rigid there must be an integer $\boldsymbol{k}$ such that:

$$A_{\boldsymbol{k}}^c = \{\langle x, k\rangle \mid \langle x, k\rangle \in A^c \text{ and } k \leq \boldsymbol{k}\}$$

is P-complete. By hypothesis, if we take $\bar{\boldsymbol{k}} = g(\boldsymbol{k})$, then

$$B_{\bar{\boldsymbol{k}}}^c = \{\langle y, h\rangle \mid \langle y, h\rangle \in B^c \text{ and } h \leq g(\boldsymbol{k})\}$$

contains $f(A_k^c)$ and, hence, if there was a polynomial algorithm for B_k^c it could be used to decide A_k^c in polynomial time. In fact in order to decide whether $\langle x, k\rangle$ belongs to A_k^c in the case $k \leq k$ (otherwise we trivially know that $\langle x, k\rangle$ does not belong to A_k^c), we may consider $\langle f_1, (x), f_2(x, k)\rangle$ and decide whether it belongs to B_k^c.

Remark. Note that under the same conditions of Theorem 2 if $A \leq B$ and B is simple A must be simple. This result shows that no polynomial reduction from a rigid problem to a simple problem is possible unless the function f_2 is such that for no computable function g it is true that, for every x and every k, $f_2(x, k) \leq g(k)$. In other words $f_2(x, k)$ cannot be dependent only on k but must eventually increase with respect to x.

Notice that Theorem 2 strengthens another result given in [11] where g is not an arbitrary monotone function but just a polynomial and the only considered case is when $f_2(x, k)$ is equal to $g(k)$.

When we pass from simple problems to strongly simple problems we obtain the following result.

Theorem 3. *Let A and B be two* NPCO *problems and $A \leq B$ via the reduction $f = \langle f_1, f_2\rangle$. If there exists a polynomial t such that, for all $x \in$ INPUT$_A$ and $k \in N^*$, $f_2(x, k) \leq t(|x|, k)$, then B strongly simple implies A strongly simple.*

Proof. If B is strongly simple, then for all polynomials p we know that the set B_p^c must be polynomially recognizable. Now, let us consider any polynomial r and the set

$$A_r^c = \{\langle x, k\rangle | \langle x, k\rangle \in A^c \text{ and } k \leq r(|x|)\}$$

we shall show that A_r^c is polynomially decidable. In fact, given $\langle x, k\rangle$, if $k > r(|x|)$ we immediately know that $\langle x, k\rangle$ does not belong to A_r^c. On the other side, if $k \leq r(|x|)$ let us consider the following set:

$$f(A_r^c) = \{\langle f_1(x), f_2(x, k)\rangle | \langle x, k\rangle \in A^c \text{ and } k \leq r(|x|)\}.$$

By hypothesis $f(A_r^c)$ is included in the set

$$S = \{\langle y, h\rangle | \langle y, h\rangle \in B^c \text{ and } h \leq t(|x|, r(|x|))\}.$$

Since we know that if A^c and B^c are NP-complete sets and $A^c \leq B^c$ via $\langle f_1, f_2\rangle$, then we must have $|x| \leq q(|f_1(x)|)$ for every x and a polynomial q, then there must exist a polynomial r' such that

$$B_{r'}^c = \{\langle y, h\rangle | \langle y, h\rangle \in B^c \text{ and } h \leq r'(|y|)\} \supseteq S.$$

So in order to decide whether $\langle x, h\rangle \in A_r^c$ we may use the reduction f and the polynomial algorithm that decides whether $\langle f_1(x), f_2(x, k)\rangle$ belongs to $B_{r'}^c$. Hence A_r^c is also polynomially decidable.

An interesting consequence of this fact is that, given a problem A which is not strongly simple and a problem B which is strongly simple any reduction from A to B must violate the hypothesis.

This means that in a reduction between A and B the measure must increase exponentially. If we consider similar reductions given by Karp [8] (e.g. EXACT–COVER $\leqslant$ KNAPSACK) we notice that this is the case and by Theorem 3 we may argue that no 'easier' reduction may be found.

An analogous result holds in the case of p-simple problems. First of all we prove the following lemma:

Lemma 1. *Let A be an* NPCO *problem. If A is p-simple, then, for every polynomial p, A_p^c is recognizable in $Q(|x|, p(|x|))$, where Q is a polynomial.*

Proof. Let A be p-simple. Given a polynomial p, we can decide $\langle x, k\rangle \in A_p^c$ in $Q(|x|, p(|x|))$. In fact if $k > p(|x|)$, it is obvious that $\langle x, k\rangle$ does not belong to A_p^c. Differently, we can use the following algorithmic procedure:

(1) compute $\bar{k} = p(|x|)$,

(2) decide if $\langle x, k\rangle \in A_{\bar{k}}^c$ in $Q(|x|, \bar{k})$.

The following theorem holds:

Theorem 4. *Under the same hypotheses of Theorem* 3, *B p-simple implies A p-simple.*

Proof. For every $\bar{k}$ we show that we can decide $A_{\bar{k}}^c$ in time polynomial in $|x|$ and $\bar{k}$. In fact, given $\langle x, k\rangle$, if $k \leqslant \bar{k}$ we consider $f(A_{\bar{k}}^c)$ which is included in the set $S = \{\langle y, h\rangle | \langle y, h\rangle \in B^c$ and $h \leqslant t(|x|, \bar{k})\}$. Furthermore if we consider the polynomial $r(u, \bar{k}) = t(q(u), \bar{k})$, where t and q are as in Theorem 3, B_r^c contains S and, by the lemma, B_r^c is decidable in time $Q(|y|, r(|y|, \bar{k}))$. Using the reduction f and the property of B_r^c we may decide whether $\langle x, k\rangle \in A_{\bar{k}}^c$ within time

$$Q(|f_1(x)|, t(q(|f_1(x)|), \bar{k})) = Q(p(|x|), t(q(p(|x|)), \bar{k}))$$

(due to the polynomiality of the reduction f) what means that the decision time is bounded by a polynomial in $|x|$ and $\bar{k}$.

Since no example is known of a problem which is strongly simple and not p-simple no application of Theorem 4 can be provided which is different from the application given at the end of Theorem 3.

As a conclusion of this paragraph we may observe that the results provided insofar have a twofold implication. On one side they can be used in order to characterize the computational complexity of one problem with respect to the given definitions, on the other side they establish conditions on the type of reductions that can be found among problems belonging to different classes, such as those discussed at the end of Theorem 2 and Theorem 3. As a further example we may observe that in the case of the reduction from KNAPSACK to MAX–CUT the existence of a much more succinct reduction than the one given by Karp is ensured by noting that the first problem is strongly simple while the second one is weakly rigid.

4. Strong NP-completeness and its relation to rigidity

In the preceding paragraph we have seen that in some cases the characterization of a problem B that is not fully approximable comes out of the fact that we can reduce an NP-complete combinatorial problem A^c into a subset of B^c in which the measure is bounded by a polynomial. Garey and Johnson give another way of considering subsets of the set INPUT of a problem to study the different characteristics of NPCO problems. Their paper [4] is an attempt to understand the different roles that numbers play in NPCO problems. Let us first consider, for example, the problem MAX–CUT that is a well-known NPCO problem. If we restrict to those graphs with unitary weights we obtain a seemingly easier problems SIMPLE–MAX–CUT, that, however, is still an NPCO problem. In the case of the problem JOB–SEQUENCING–WITH–DEADLINES the situation is different: it is NP-complete but if we restrict to the case when all weights are unitary, then the problem is solvable in $O(n \lg n)$. Moreover, if the weights are at most k the problem is solvable by an algorithm using a classical dynamic approach, whose complexity time is bounded by a polynomial in k and in n (the number of jobs). Note that this algorithm is not polynomial: in fact a polynomial algorithm should solve JOB–SEQUENCING–WITH–DEADLINES in time bounded by a polynomial in n and in $\lg k$.

In order to formalize these observations Garey and Johnson introduce another function of the input, $\mathrm{MAX}:\mathrm{INPUT}\to N$ that captures the notion of the magnitude of the largest number occurring in the input. For example given a weighted graph G, $\mathrm{MAX}(G)$ can be defined as the value of the maximum weighted edge.

The following definitions formalize these concepts.

Definition 9. A *pseudo-polynomial algorithm* is an algorithm that on input x runs in time bounded by a polynomial in $|x|$ and in $\mathrm{MAX}(x)$.

Definition 10. An NPCO problem is a *pseudo-polynomial NPCO problem* if there is a pseudo-polynomial algorithm that solves it.

Definition 11. Given a problem P let P_q denote the problem obtained by restricting P to only those instances x in INPUT_P for which $\mathrm{MAX}(x) \leq q(|x|)$. An NPCO problem P is *NP-complete in the strong sense* if there exists a polynomial q such that P_q is NP-complete.

According to the preceding observations an example of pseudo-polynomial NPCO problem is JOB–SEQUENCING–WITH–DEADLINES [9] while MAX–

CUT is NP-complete in the strong sense (it is sufficient to consider the constant polynomial $q(x)=1$ to obtain SIMPLE–MAX–CUT).

The two classes of pseudopolynomial NPCO problems and of strong NP-complete problems are disjoint (obviously unless P = NP). The following proposition states the relationship between strong NP-completeness and full approximability.

Proposition 2. *If P is* NP-*complete in the strong sense, then it is not fully approximable, provided that* $m^*(x)-\tilde{m}(x)\leq p(|x|, \mathrm{MAX}(x))$.

Garey and Johnson give another result that connects the two concepts of pseudopolynomial and fully approximable NPCO problem; for clarity sake, we will give it later as an immediate consequence of Theorem 6.

In many problems the optimal value of the solution and the MAX of the input are strictly related in the sense that it is possible to establish a polynomial relation between them. This suggests the idea of comparing some of the different concepts introduced in the preceding paragraphs and in this one. First of all we have the following results:

Theorem 5. *Let A be a pseudopolynomial optimization problem. If there exists a polynomial q such that, for every* $x\in \mathrm{INPUT}_A$, $\mathrm{MAX}(x)\leq q((m^*(x)-\tilde{m}(x)), |x|)$, *then A is p-simple.*

Proof. The hypotheses imply that there exists a polynomial p such that, given x, $m^*(x)$ is computable within time $p(|x|, \mathrm{MAX}(x))$ and, therefore, within time $p(|x|, q((m^*(x)-\tilde{m}(x)), |x|))$. Given k we can decide whether $\langle x, k\rangle\in A_k^c$ by applying the pseudopolynomial algorithm for $p(|x|, q(k, |x|))$ steps. If the algorithm stops we know $m^*(x)$ and therefore we can decide if $m^*(x)-\tilde{m}(x)\geq k$ or $m^*(x)-\tilde{m}(x)<k$; otherwise $m^*(x)-\tilde{m}(x)>k$ and hence $\langle x, k\rangle\in A_k^c$.

Theorem 6. *Let A be a p-simple problem. If there exists a polynomial q such that, for every* $x\in\mathrm{INPUT}_A$, $m^*(x)-\tilde{m}(x)\leq q(\mathrm{MAX}(x), |x|)$, *then A is a pseudopolynomial* NPCO *problem.*

Proof. By hypothesis, for each k, A_k^c is recognizable in time $Q(|x|, k)$. To obtain $m^*(x)$ we can use the following algorithm:

find $\tilde{m}(x)$
for $k:=1$ to $q(\mathrm{MAX}(x), |x|)$
repeat the following step:
if $\langle x, k\rangle\in A_k^c$; then $m^*(x)=\tilde{m}(x)+k$.

There are no more than $q(\mathrm{MAX}(x), |x|)$ iterations and as A^c is p-simple, each iteration takes no more than $Q(|x|, q(\mathrm{MAX}(x), |x|))$. Therefore $m^*(x)$ is computable in at most $(q(\mathrm{MAX}(x), |x|))\cdot Q(|x|, q(\mathrm{MAX}(x),|x|))$ steps.

Corollary 1. [4]. *Let A be a fully approximable* NPCO *problem. If there exists a polynomial q such that for every* $x \in \mathrm{INPUT}_A$ $(m^*(x) - \tilde{m}(x)) \leq q(\mathrm{MAX}(x), |x|)$, *then A is a pseudopolynomial* NPCO *problem.*

Proof. Immediate from the previous theorem and the fact that a fully approximable problem is p-simple.

As the conditions of Theorem 5 and 6 are generally verified the two concepts of pseudopolynomial problem and p-simple problem coincide in many cases.

A natural question arises at this point: when the conditions of the theorems are not verified which of the two approaches gives a better information about the complexity of approximate algorithms?

Let us define:

$$\text{(P1)} \quad \begin{array}{ll} \text{Max} & \sum_{j=1}^{n} c_j y_j \\ \text{s.t.} & \sum_{j=1}^{n} a_j y_j = b; \quad y_j = 0,1; \quad c_j, a_j > 0; \quad j = 1, 2, \ldots, n. \end{array}$$

Since $\mathrm{INPUT}_{\mathrm{P1}}$ is $\langle (a_1, c_1), (a_2, c_2), \ldots, (a_n, c_n); b \rangle$, a natural definition of $\mathrm{MAX}(\mathrm{INPUT}_{\mathrm{P1}})$ can be the following $\mathrm{MAX}(x) = \max_j(c_j, a_j)$ and it is not hard to prove that P1 is pseudopolynomial (a classic dynamic approach solves it in $\mathrm{O}(n^2\mathrm{MAX}(x))$); however, the problem to obtain a solution in the case that all c_j are equal to 1 is an NP-complete problem [8] and therefore P1 is weakly rigid. Hence P1 is a pseudopolynomial NPCO problem that is weakly rigid and not fully approximable.

Let us consider now:

$$\text{(P2)} \quad \begin{array}{ll} \text{Max} & \prod_{j=1}^{n} c_j^{y_j} \\ \text{s.t.} & \prod_{j=1}^{n} a_j^{y_j} \leq 2^b; \qquad y_j = 0, 1; \qquad c_j, a_j > 0; \qquad j = 1, 2, \ldots, n. \end{array}$$

In this case $\mathrm{INPUT}_{\mathrm{P2}}$ is $\langle (a_1, c_1), (a_2, c_2), \ldots, (a_n, c_n); b \rangle$; this problem is fully approximable and therefore p-simple; we conjecture that it is not a pseudopolynomial problem because the classical method of deriving a pseudopolynomial algorithm from the dynamic programming approach does not work.

We have proven that the two approaches are equivalent under restricted but reasonable hypotheses and we have shown, in the previous examples, that, when $m^*(x)$ and $\mathrm{MAX}(x)$ are not polynomially related the approach formulated by Paz and Moran has a wider application to study approximation properties for NPCO problems. Furthermore, it seems to us that their approach is straightforward while, in some cases, the definition of the function MAX may be ambiguous as the above example shows.

Before finishing this paragraph we want to observe that, when there is a polynomial relation between the value of the optimal solution and the value of MAX, there is a strong connection between the two concepts of strong NP-complete and weakly rigid.

Theorem 7. *Let A be a strong* NP-*complete optimization problem. If there exists a polynomial p such that, for every $x \in \mathrm{INPUT}_A$, $(m^*(x) - \tilde{m}(x)) \leq p(\mathrm{MAX}(x), |x|)$, then A is weakly rigid.*

Proof. If A is NP-complete in the strong sense there must exist a polynomial q such that the set $Q = \{\langle x, k\rangle \mid \langle x, k\rangle \in A^c, \mathrm{MAX}(x) \leq q(|x|)\}$ is NP-complete.

Let us consider now the set

$$Q' = \{\langle x, k\rangle \mid \langle x, k\rangle \in A^c, \mathrm{MAX}(x) \leq q(|x|), k \leq p(\mathrm{MAX}(x), |x|)\}.$$

As $Q \supseteq Q'$ in order to prove that $Q \equiv Q'$ it is sufficient to prove that

$$Q - Q' = \{\langle x, k\rangle \mid \langle x, k\rangle \in A^c, \mathrm{MAX}(x) \leq q(|x|), k > p(\mathrm{MAX}(x), |x|)\}$$

is the empty set. In fact given $\langle x, k\rangle$, with $k > p(\mathrm{MAX}(x), |x|)$, we have by hypothesis $k > p(\mathrm{MAX}(x), |x|) \geq m^*(x) - \tilde{m}(x)$ and therefore $\langle x, k\rangle \notin A^c$. Let us consider now

$$Q'' = \{\langle x, k\rangle \mid \langle x, k\rangle \in A^c, k \leq p(q(|x|), |x|)\}.$$

Since Q' is NP-complete, clearly Q'' is NP-complete and hence A is weakly rigid.

5. Conclusions

In this paper we have shown that there exist close relations among different approaches to the classification of NP-complete optimization problems, giving also new results on the type of possible reductions among problems belonging to different classes. On the other side, it was proven that, under natural conditions, various classifications of NP-complete optimization problems are actually equivalent and that violating these conditions this equivalence does not hold anymore.

From a general point of view our paper and the others quoted in the preceding paragraphs strengthen our conviction that in order to provide meaningful characterizations of NPCO problems it is necessary to find the suitable level of abstraction because if a too general point of view is taken NPCO problems appear to be hardly distinguishable while if too many details are taken into consideration it is difficult to grasp similarities among different problems.

The study that we started in this paper is an attempt in this direction. Along the same line we think that in order to establish connections among combinatorial structure, complexity and approximation properties of NPCO problems it may be fruitful to find relationships with other results drawn from other approaches, at the same level of abstraction, such as the one developed in [1].

References

[1] G. Ausiello, A. D'Atri and M. Protasi, On the structure of combinatorial problems and structure preserving reductions, *Proc. 4th International Colloquium on Automata, Languages and Programming,* Turku (1977).

[2] G. Ausiello, A. D'Atri and M. Protasi, Lattice theoretic ordering properties for NP-complete optimization problems, *Fund. Informat.*, to appear.

[3] M.R. Garey and D.S. Johnson, Approximation algorithms for combinatorial problems: an annotated bibliography in: J.F. Traub, Ed., *Algorithms and Complexity: New Directions and Recent Results* (Academic Press, New York, 1976).

[4] M.R. Garey and D.S. Johnson, "Strong" NP-completeness results: motivation, examples and implications, *J. ACM* **25** (3) (1978).

[5] E.M. Gold, Limiting recursion, *J. Symbolic Logic* **30** (1) (1965).

[6] J. Hartmanis and L. Berman, On isomorphism and density of NP and other complete sets, *Proc. 8th ACM Symposium on Theory of Computing* (1976).

[7] D.S. Johnson, Approximation algorithms for combinatorial problems, *Proc. 5th ACM Symposium on Theory of Computing* (1973).

[8] R.M. Karp, Reducibility among combinatorial problems, in: R.E. Miller and J.W. Thatcher, Eds., *Complexity of Computer Computations* (Plenum Press, New York, 1972).

[9] E.L. Lawler and J.M. Moore, A functional equation and its application to resource allocation and sequencing problems, *Management Sci.* **16** (1) (1969).

[10] S. Moran, Efficiently approximable problems: characterizations and examples, Technical Report 135, Department of Computer Science, Haifa (1978).

[11] A. Paz and S. Moran, NP-optimization problems and their approximation, *Proc. 4th International Colloquium on Automata Languages and Programming,* Turku (1977).

[12] S. Sahni, Algorithms for scheduling independents tasks, *J.ACM* **3** (1) (1976).

[13] F. Ventriglia, A. Aiello, E. Burattini and A. Massarotti, A new evaluation function for approximation algorithms, *Proc. Informatica 77*, Bled (1977).

[14] E. Zemel, Functions for measuring the quality of approximate solutions to zero-one programming problems, Technical Report No. 323, Graduate School of Management, Northwestern University (June 1978).