FULL APPROXIMABILITY OF A CLASS OF PROBLEMS OVER POWER SETS

G. AUSIELLO, A. MARCHETTI SPACCAMELA, M. PROTASI

In this paper results concerning structural and approximability properties of the subclass of NP-Complete Optimization problems, defined over a lattice are considered. First, various approaches to the concept of Fully Polynomial Approximation Scheme are presented with application to several known problems in the class of NP-Complete Optimization Problems.

Secondly, a characterization of full Approximability for the class of Max ---- Subset Problems is introduced.

1. INTRODUCTION

The aim of this paper is to discuss methods for the full approximation of combinatorial problems and to study the full approximability of a class of NP-complete optimization -- problems defined over a lattice. Most combinatorial optimization problems can be naturally defined as optimization problems over -- lattices according to the ground algebraic - structure of the set of feasible solutions. For example the problem of graph colouring - can be viewed as an optimization problem --- over a partition lattice, the problem of minimum spanning trees is an optimization problem over a matroid, ecc. In /1 / a large - class of optimization problems was formalized as the class of max-subset problems over --- power sets and some basic properties of these problems were studied. Despite its simple - characterization the class of max-subset problems is indeed sufficiently general to in-- clude problems with very different properties with respect to approximability. In fact - this class includes many problems which are known to be non fully approximable and, at -- the same time, practically all known examples of fully approximable NP-complete problems.

The existence of good approximations to the solution of hard optimization problems has - been studied by several authors /8/,/2/,/4/ /5/, etc.).

What is more interesting for the development of our work is that 1) the techniques used - in proving the full approximability of a problem are essentially based on variations of dynamic programming, 2) generally single problems (and not classes of problems) have been shown to be fully approximable.

In particular as regards 2) many difficulties - arise when trying to find general natural -- conditions for the approximability and des-- pite of the interest for this type of results few steps have been made in this direction - (/7/, /13/).

In order to establish a connection between -- good approximability of hard problems and -- the intrinsic combinatorial properties which characterize such problems it is useful to - restrict ourselves to considering max-subset problems and the properties of the set of -- their feasible solutions.

In the whole we can say that three possible research areas are worth-while of beeing pursued: 1) to find new simple methods of full - approximation, 2) to give general conditions for the full approximability of a class of --

G. Ausiello - Istituto di Automatica, Università di Roma and Centro di Studio dei Sistemi di Controllo e Calcolo Automatici del C.N.R., Roma.
A. Marchetti Spaccamela- Centro di Studio dei Sistemi di Controllo e Calcolo Automatici del C.N.R., Roma.
M. Protasi - Istituto Matematico, Università dell'Aquila, L'Aquila.

problems, 3) to introduce new approximate algorithms of lower complexity for problems -- which are already known to be fully approximable.

In this paper we will be concerned with ---- points 1 and 2). In fact, in 3. we will consider a new method for showing the full - approximability. Its computational complexity will be studied and its advantages with - respect to the classical schemes will be also shown. Instead in 4. we will give a sufficient condition for the full approximability of a subclass of max-subset problems which is based on the structural properties of the set of feasible solutions and which - is verified by the most important problems - which are known to be fully approximable.

2. A FULLY POLYNOMIAL APPROXIMATION SCHEME

Given an NP-complete optimization problem A with measure m the following definitions capture the concept of good approximability.

DEFINITION 2.1. A is an ε-*approximate algorithm* for A if, given any instance $x \in A$, we have

$$\left| \frac{m^*(x) - m(A(x))}{m^*(x)} \right| \leq \varepsilon .$$

where $m^*(x)$ is the measure of the optimal -- solution of the instance x.

DEFINITION 2.2. A problem A is said to be

a) *polynomially approximable* if given any $\varepsilon > 0$ there exists an ε-approximate algorithm for A which runs in polynomial time; ---

b) *fully polynomially approximable* if A is polynomially approximable and there exist - a polynomial q such that; given any ε, the running time of the ε-approximate algorithm is bounded by $q(|x|, 1/\varepsilon)$.

DEFINITION 2.3. A constructive method that for any given ε provides the corresponding polynomial ε-approximate algorithm A_ε is - said to be a *polynomial approximation scheme* (PAS). Besides if, for every ε, the running time of A_ε is bounded by $q(|x|, 1/\varepsilon)$ for - some polynomial q we say that the scheme is a *fully polynomial approximation scheme*.

As we said in the introduction the main aim of this paper is to characterize optimiza--- tion problems belonging to the class of max-subset problems which are fully approximable and hence we will only consider fully polynomial approximation schemes for this class of problems.

DEFINITION 2.4. A NP *max-subset problem* A - over an alphabet Σ is a quadruple

$A = \langle \text{INPUT}, F, \pi, m \rangle$ where

INPUT: is a polynomially decidable subset of Σ^* (set of *instances*)

F : INPUT $\to$ P $P(\Sigma^*)$ is a polynomially computable mapping that to every input x associates a finite set of (encodings of) *objects*

π : is a polynomially decidable *property* of subsets of F(x)

m : F(INPUT) N(where $F(\text{INPUT}) = \bigcup_{x \in \text{INPUT}} P(F(x))$) $\longrightarrow$ is the *measure* that, given x, asso--- ciates a non negative integer to ---- every subset of F(x).

DEFINITION 2.5. Given an instance x of a NP max-subset problem A,

1) the *search space* of x is the lattice L_x of the powerset P(F(x)) under inclusion
2) the set of *feasible solutions* of x is the subsemilattice SOL(x), which is formed by elements of L_x which satisfy π
3) the *optimal solutions* of x are the ele--- ments of SOL(x) for which m is maximal.

The definitions can be extended to minimization problems by inverting the lattice ordering.

Examples : graph problems = Max-clique, Min node cover, node deletion, arc - deletion, max-subgraph
: set problems = Max-set packing,- min-set covering, min hitting -- set
: mathematical programming = Max-Knapsack, max subset sum
: problems of scheduling theory
: problems on matroids and independent systems

(For the definitions of the above problems see /2/).

As a detailed example let us consider the - 0-1 knapsack problem

$\max \sum_i c_i x_i$ subject to $\sum_i a_i x_i \leq b \quad x_i \in \{0,1\}$

In this case we have:

INPUT = (2n+1)-tuples of positive inetgers

$\langle c_1,\ldots,c_n;\ a_1,\ldots,a_n;b \rangle$

$F(\langle c_1,\ldots,c_n;\ a_1,\ldots,a_n;\ b\rangle) = \{c_1,\ldots,c_n\}$

$\pi(\{c_{j_1},\ldots,c_{j_K}\}) \Leftrightarrow a_{j_1} + \ldots + a_{j_K} \leq b$

$m(\{c_{j_1},\ldots,c_{j_K}\}) = \sum_{i=I}^{K} c_{j_i}$

The fundamental technique for constructing - fully polynomial approximation schemes are - all based on the classic dynamic programming scheme. This scheme, in the case of max sub set problems can be summarized

```
L:= ∅;
for all items i in F(x) do
        for all sets S_j in L do
            if S_j ∪ {i} satisfies π
            then
              begin insert S_j ∪ {i} in L;
                    eliminate dominated elements
              end
        end
    for
take the best solution in L.
```

It is easy to see that the number of steps - of the algorithm is proportional to the number of items in F(x) times the lenght of the list L.

Clearly variations of this scheme are obtained by considering different conditions of -- dominance between elements.

In the case of knapsack we can define the following dominance rule:

Given two sets S' and S" in L we say that S' is *dominated* by S"

if $\sum_{i\in S'} c_i c_i \leq \sum_{i\in S"} c_i$ and

$\sum_{i\in S'} a_i a_i \leq \sum_{i\in S'} a_i$

Clearly the elimination of S' does not introduce any error.

Therefore we can obtain the following exact algorithm for the knapsack problem:

Algorithm A_1

```
L: ∅;
for i = 1 to n do
    for all sets S_j in L do
        if  Σ_{j∈S_j} a_j + a_i ≤ b
            then
               beging L: = L∪(S_j ∪ {i})
                     eliminate all S' ∈ L
                     such that ∃ S" ∈ L
                     Σ_{j∈S'} c_j ≤ Σ_{j∈S"} c_j
                     and
                     Σ_{j∈S'} a_j ≥ Σ_{j∈S"} a_j
               end
    end
end
take the best solution in L.
```

To evaluate the complexity of the above algorithm it is sufficient to see that, at each step, the number of solutions contained in - the list L is less than

$$\min\left(b, \sum_{j=1}^{n} a_j, \sum_{j=1}^{n} c_j\right).$$

So with a suitable implementation of the elimination step it is not hard to see that the complexity of algorithm A_1 is

$O(n \cdot \min(b, \sum_{j=1}^{n} a_j, \sum_{j=1}^{n} c_j))$, which means a - complexity exponential in the size of the input, as we use a binary encoding for the numbers of the input.

It is also possible to obtain the elements of the optimal solution without increasing the overall complexity of the algorithm (see /5/).

In order to achieve a fully polynomial approximation scheme the first technique which -- was used for finding an approximate solution to the knapsack problem was based on scaling all coefficients a_i by a factor $K = \varepsilon \cdot a_{MAX}/n$.

This technique is summarized by the following algorithm

Algorithm A_2

```
for i = 1 to n do
  c'_j = k·c_i
end;
Apply algorithm A_1 taking as input
(c'_1...,c'_n; a_1...,a'_n;b)
take the best solution and multiply
it for k.
```

If $m(A_2(x))$ is the value of the approximate solution we have that

$$m^*(x) - m(A_2(x)) \leq n \cdot k$$

On the other side we can assume (w.l.o.g.) that

$$m^*(x) \geq c_{MAX} \cdot$$

it follows that

$$\frac{m^*(x) - m(A_2(x))}{m^*(x)} \leq \frac{n \cdot k}{c_{MAX}} = \varepsilon$$

With respect to the running time we have that the complexity of the algorithm is $0(n \cdot (\sum c'_i))$. Due to the scaling we have that

$$\sum c'_i \leq \frac{n \cdot c_{MAX}}{k} = \frac{n^2}{\varepsilon}$$

So the overall complexity is $0(\frac{n^3}{\varepsilon})$.

Algorithm A_2 can be improved in several different ways obtained by Ibarra and Kim /14/ and Lawler /5/.

Ibarra and Kim use a better lower bound P based on the value G of the greedy algorithm

$$P = \max \{G, c_{MAX}\}$$

$$P \leq m^*(x) \leq 2P$$

It allows to use a bigger scaling factor $k' = \frac{\varepsilon \cdot P}{n}$.

Therefore the complexity of the algorithm is $0(n \frac{2P}{k}) = 0(\frac{n^2}{\varepsilon})$. Besides this the items are separated in small and large items, Algorithm A_1 is applied only to large items and afterwards the solutions obtained are improved using a greedy procedure.

So the computational complexity of the algorith is $0(n \lg + \frac{1}{\varepsilon^4} \lg \frac{1}{\varepsilon})$. Lawler uses a k-median finding routine to avoid the ordering of the elements to obtain the lower bound. So the complexity is lowered to $0(n \lg \frac{1}{\varepsilon} + (\frac{1}{\varepsilon^4}) \lg 1/\varepsilon)$. Finally a variable scaling factor (the larger is the profit of the element the larger is the scaling factor) allows to get $0(n \lg \frac{1}{\varepsilon} + \frac{1}{\varepsilon^4})$.

3. DIFFERENT FULLY POLYNOMIAL APPROXIMATION SCHEMES

The fully polynomial approximation scheme described in Par. 2, although very useful for many problems, suffers some drawbacks.

In fact in order to find the fully polynomial approximation scheme we need to know good bounds to m^* and this is a severe limitation to the generality of the method as it can be easily seen if we simply switch from max knapsack to min knapsack problems.

Another limitation of this scheme is that it cannot be applied for solving other NP-complete optimization problems which instead can be shown to be fully approximable by other methods such as the product knapsack problem.

Due to these facts the search for general full approximation schemes has been pursued with the aim of finding results which, despite a slight loss in efficiency could be applied to a broader class of problems and that could provide some insight in the properties of fully approximable problems and in their characterization.

The first attempt to provide such a general scheme was the condensation algorithm due to Moran /6/. With respect to the dynamic programming schema (A_1) the elimination step is performed by eliminating more partial solutions and therefore introducing an error.

More precisely we say that S" dominates S'

if $(1-\delta) \sum_{i \in S'} c_i \leq \sum_{i \in S''} c_i$ and

$$\sum_{i \in S'} a_i \geq \sum_{i \in S''} a_i$$

where $\delta = \min\{\varepsilon^2, \frac{1}{n^2}\}$, the condensing parameter, is the relative error introduced in the dominance test. As there is a propagation of the error then the total relative error is at least $\delta^2 \leq \varepsilon$. Moreover the running time, as analyzed by Moran, is $0(\max\{|x^4|, |x^2|/\varepsilon^2\})$ when applied to variants of max subset sum and max subset-product problems.

A different approach which leads to a more efficient algorithm is is based on the technique

of *variable partitioning* (as opposed to the constant partitioning technique introduced - by Sahni /8/). This method is based on the partitioning of the range of the measure into intervals of exponentially increasing --- size and on an elimination rule which preserves only one solution for every interval.

To allow a better understanding of the advantages of this approach the method and the results will be given for the 0/1 knapsack - and the 0/1 product knapsack. It can be immediately extended to other fully approximable problems.

More in detail the method is as follows.

Let R be the range of the possible values of the measure. In a general NP-complete max-subset problem, and therefore in our cases R is smaller than $2^{P(|x|)}$ for some polynomial P and as we will see the whole development - of the algorithm allows us to refer only to this general bound without requiring any more precise estimate of a bound for m^*. The --- range R is then partitioned into K intervals $[0,m_1),[m_1,m_2),\ldots [m_{K-1},m_K)$ where $m_i = (1+\varepsilon/n)^i$. Let us denote T_i the i-th interval.

The elimination rule for the 0/1 knapsack is the following:

Given two sets S' and S", S' is *dominated* by S" if

$$\text{if } \sum_{i\in S'} c_i \in T_i,\ \sum_{i\in S"} c_i \in T_j,\ j \geq i \text{ and } \sum_{i\in S'} a_i \geq \sum_{i\in S"} a_i$$

Clearly changing the sums in products we --- have the elimination rule for the 0/1 product knapsack.

In every interval there will be at most one feasible solution and hence, at each iteration, we will have, at most R elements in -- the list.

THEOREM 2.1. The variable partitioning method provides a fully polynomial approximation scheme for the 0/1 knapsack and the 0/1 product knapsack.

PROOF. The error that may result by using - this algorithm may be bounded as follows. At stage i at most the error $\Delta_i = m_i - m_{i-1}$ may -- arise; in the worst case this error may happen at every stage. Since there are n stages and since $\Delta_i < \Delta_{i+1}$ we have that

$$|m^*(x) - m(A_\varepsilon^*(x))| \leq n\,\Delta_{i_{MAX}}$$

where i_{MAX} is such that $m_{i_{MAX}-1} \leq m^*(x) < m_{i_{MAX}}$ From the above inegualities we deduce that - the overall error is

$$\left|\frac{m^*(x)-m(A_\varepsilon(x))}{m^*(x)}\right| \leq \frac{n\left[(1+\frac{\varepsilon}{n})^{i_{MAX}} - (1+\frac{\varepsilon}{n})^{i_{MAX}-1}\right]}{(1+\frac{\varepsilon}{n})^{i_{MAX}-1}} = \varepsilon$$

As far as the complexity is concerned, the number of steps of the given algorithm is as usual a function of n and the lenght of the list L. In this case the number of solutions which may be preserved in L is equal - to the number of intervales K which should - satisfy the following inequalities.

$$(1+\tfrac{\varepsilon}{n})^K \leq 2p(|x|) \qquad K\log(1+\tfrac{\varepsilon}{n}) \leq p(|x|)$$

$$K \leq \frac{p(|x|)}{\log(1+\frac{\varepsilon}{n})}$$

Hence with a suitable implementation the -- complexity of the method is

$$0\left(n.\frac{p(|x|)}{\log(1+\frac{\varepsilon}{n})}\right)$$

Therefore in the case of knapsack we have - that the range R is bounded by $n\cdot a_{MAX}$ and therefore in this case we have a complexity

$$0\left(n.\ \frac{\log n + \log a_{MAX}}{\log(1+\varepsilon/n)}\right)$$

while in the case of product knapsack

$$0\left(n^2.\ \frac{\log a_{MAX}}{\log(1+\varepsilon/n)}\right)$$

QED

The complexity of the method could be improved in two directions; a) from a general -- point of view using together the variable - partitioning with Sahni's fixed partioning b) for a single problem, exploiting some -- particular features. For instance some --- ideas by Ibarra, Kim and Lawler for the ---

Knapsack could also be applied in our case.

However we will not describe these results further because they are obvius extensions and because in this paper we are interested in the general characteristics of the scheme and in defining conditions which guarantee its applicability.

4. A SUFICIENT CONDITION FOR THE FULL APPROXIMABILITY OF MAX SUBSET PROBLEMS

The results shown in the preceding paragraph suggest to introduce an abstract characterization of the condition of dominance that allows the elimination of feasible solutions and to establish on this basis a condition of full approximability for max-subset problems.

For this purpose we have to require that the satisfaction of the property π by a feasible solution of a max subset problem is "measured" by a function f (which generalizes the concept of occupancy as it appears in knapsack problems).

DEFINITION 3.1. A max subset problem *A* is said to be *regular* if there exists a polynomially computable set function f with integer value such that the following conditions hold:

1. for every $S \in P(F(x))$ $f(S) \leq 0$ iff $\pi(S)$

2. $\forall S \quad f(\phi) \leq f(S) \quad , \; m(\phi) \leq m(S)$

3. $\forall S_1, S_2$ and any disjoint S_3

   $f(S_1) \leq f(S_2) \rightarrow f(S_1 \cup S_3) \leq f(S_2 \cup S_3)$

   $m(S_1) \leq m(S_2) \rightarrow m(S_1 \cup S_3) \leq m(S_2 \cup S_3)$

4. $\forall S_1, S_2$ and disjoint S_3

   $$m(S_1) \geq m(S_2) \rightarrow \frac{m(S_1)}{m(S_2)} \geq \frac{m(S_1 \cup S_3)}{m(S_2 \cup S_3)}$$

In /1/ two weaker properties of max-subset problems were introduced, namely the *hereditarity* of the property

$$\forall S_1, S_2 \quad S_1 \subseteq S_2 \rightarrow (\pi(S_2) \rightarrow \pi(S_1))$$

and *monotonicity* of the measure

$$\forall S_1, S_2 \quad S_1 \subseteq S_2 \rightarrow m(S_1) \leq m(S_2)$$

PROPOSITION 3.1. A *regular* max-subset problem has the properties of hereditarity and monotonicity.

PROOF. Let $S_1 = \phi$.

Since property 2 of definition 3.1 guarantees $f(\emptyset) \leq f(S_2)$ and $m(\emptyset) \leq m(S_2)$ for every S_2; then, given any S_2 and S_3, property 3 of definition 3.1 implies

$m(S_3) \leq m(S_2 \cup S_3)$ (monotonicity)

and $f(S_3) \leq f(S_2 \cup S_3)$

that is $\pi(S_2 \cup S_3) \rightarrow \pi(S_3)$ (hereditarity)

QED

On the other side the property of regularity is indeed strictly stronger than hereditarity and monotonicity.

PROPOSITION 3.2. Max-clique is hereditary and monotone but is not regular.

PROOF. The fact that max-clique is hereditary and monotone is trivial. On the other side whatever f we choose there will be instances of the problem and sets S_1, S_2, S_3 of nodes such that S_1 and S_2 are nodes of complete subgraphs, $S_1 \cup S_3$ corresponds to a complete subgraph, $S_2 \cup S_3$ corresponds to a non complete subgraph. Then we would have $f(S_1) = a$, $f(S_2) = b$ for some negative a and b. W.l.o.g. let $b \leq a$. At the same time $f(S_1 \cup S_3) = c \leq 0$ while $f(S_2 \cup S_3) > 0$. So we would have $f(S_2) < f(S_1)$ but $f(S_2 \cup S_3) > f(S_1 \cup S_3)$.

QED

Examples of problems that satisfy the definition of regularity are knapsack, product knapsack, some scheduling problems. The fact that all these problems are also known to be fully approximable is not surprising because we may prove the following theorem:

THEOREM 3.3. A regular NP-complete max-subset problem is fully approximable.

PROOF. Let us consider the following algorithm based on Moran's approach

$\delta = \min(\frac{1}{n}, \varepsilon)$;

L: = ∅
for all items i in F(x) *do*
for all sets S_j in L *do*
if $S_j \cup \{i\}$ satisfies π
then
begin insert $S_j \cup \{i\}$ in L; {elimination step} eliminate all elements $S' \in L$ for which there exists $S'' \in L$ such that

$(1 - \frac{\delta}{n}) m(S') \leq m(S'')$ and

$f(S') \geq f(S'')$

end
end
end
take the best solution in L.

As regards the analysis of the error we observe that, at each step, the error introduced by eliminating S and keeping S is at most $\frac{\delta}{n}$. By property 3 of definition 3.1 we have that for each subset T (disjoint with S' and S") if $S' \cup T$ is feasible then also $S'' \cup T$ is feasible. By property 4 we have that if $m(S'') \leq m(S')$ then

$$\frac{m(S'' \cup T)}{m(S' \cup T)} \geq \frac{m(S'')}{m(S')} \geq 1 - \frac{\delta}{n} ;$$

in the other case by property 3 we have that $m(S'' \cup T) \geq m(S' \cup T)$.

Hence there is a propagation of the error introduced at each step; since there are n steps the total error is $n \cdot \frac{\delta}{n} \leq \varepsilon$. The computational complexity of the algorithm is, with a suitable implementation, $0(n \ell)$ when ℓ is the maximum number of partial solutions in the list L. As there exists a polynomial $p(|F(x)|)$ such that $m(F(x)) \leq 2^{p(|F(x)|)} = 2^{p(n)}$; we have that $\ell = 0(\lg_{\delta} F(x)) = 0(\frac{\ln F(x)}{\delta}) = 0(\frac{p(|F(x)|)}{\delta})$.
Therefore the complexity of the algorithm is

$0\ (p(n) \max(n^2, \frac{n}{\varepsilon}))$.

QED

5. BIBLIOGRAFIA

/1/ AUSIELLO,G., MARCHETTI SPACCAMELA, A, PROTASI, M. : "Combinatorial problems over power sets, Calcolo, Vol. 16, n.4, 1979.

/2/ GAREY, M.R., JOHNSON, D.S. :"Computers and intractability. A guide to the ---- theory of NP-completeness", Freeman and Company, 1979.

/3/ KORTE, B. : "Personal communication".

/4/ IBARRA, O.H., KIM, C.E. : "Fast approximation algorithms for the knapsack -- and sum subset problems, J.ACM, Vol 22, n.4, 1975.

/5/ LAWLER, E.L. : "Fast approximation algorithms for knapsack problems", Proc, 18th FOCS, Long Beach, 1977.

/6/ MORAN, S. : "General approximation algorithms for some arithmetical combinatorial problems", Tech. Rep. 140, Technion, Haifa, 1978.

/7/ PAZ, A., MORAN, S. : "NP-optimization problems and their approximations", --- Proc. 4th ICALP, Turku, 1977.

/8/ SAHNI, S. : "Approximate algorithms -- for the 0/1 knapsack problem", J.ACM. - Vol. 22, n.1, 1975.