Computer Science Journal of Moldova, vol.**12**, no.3(36), 2004

An algorithm for solving a class of knapsack problems and its generalization

Elena Balan

Abstract

A class of knapsack problems which generalizes the classical ones is studied. Algorithms based on the dynamical programming and Branch and Bound methods are proposed. The correctness and time estimation of the algorithm are given.

1 Problem formulation

We study a class of knapsack problems which generalizes problems from [1-2]. The main mathematical model we shall use is formulated as follows.

Let a knapsack of size D and a set of items $I = \{1, 2, \ldots, n\}$ be given. For each item $j \in I$ the cost c_j, the size d_j and the volume d'_j related to partition I are known. The set I is divided into p non-empty disjoint subsets $I_1, I_2, \ldots, I_p$,

$$I = I_1 \cup I_2 \cup \cdots \cup I_p;\ I_i \cap I_j = \emptyset,\ i \neq j.$$

For each subset I_l, $l = \overline{1,p}$, the size D_l is known.

According to this partition, from each of the subsets I_l, $l = \overline{1,p}$, we include into the knapsack no more than k_l items and the sum of volumes d'_j of these including items from I_l does not exceed D_l. A sequence of items $i_1, i_2, \ldots, i_q$ $(q < n)$ from I we name the packing in the knapsack if the sum of items sizes d_j of the sequence does not exceed D, each of the subsets I_l, $l = \overline{1,p}$, contains at most k_l items of the sequence and the sum of items volumes related to the partition d'_j

for each I_l does not exceed D_l, $l = \overline{1,p}$. We consider the problem of finding of the packing in the knapsack which has the maximal sum of items costs.

This problem can be represented as a boolean linear programming model, which in the case $p = n$, $D_l = D$ and $k_l = 1$, $l = \overline{1,p}$, becomes the classical knapsack problem. Therefore it is NP-complete. We propose an algorithm of dynamical programming method for solving the formulated problem.

In the terms of boolean programming the problem can be formulated as follows.

To maximize the object function

$$\mu(x) = \sum_{j \in I} c_j x_j$$

on the subject

$$\begin{cases} \sum\limits_{j \in I} d_j x_j \le D; \\ \sum\limits_{j \in I_l} d'_j x_j \le D_l, \ l = \overline{1,p}; \\ \sum\limits_{j \in I_l} x_j \le k_l, \ l = \overline{1,p}; \\ x_j \in \{0,1\}, \ j \in I, \end{cases}$$

where $x = (x_1, x_2, \ldots, x_n)$. Here $x_j = 1$ if the item j is included into the knapsack; otherwise $x_j = 0$.

2 The main results and algorithms for solving the problem

To solve this problem we shall use an algorithm for solving the following auxiliary problem:

to maximize the object function

$$\mu(x) = \sum_{j \in I_l} c_j x_j$$

on the subject

$$\begin{cases} \sum\limits_{j\in I_l} d'_j x_j \le D_l; \\ \sum\limits_{j\in I_l} x_j \le k_l; \\ x_j \in \{0,1\},\ j \in I_l, \end{cases}$$

We will number the elements of I_l as $1_l, 2_l, \ldots, n_l$.

Algorithm 1

1. Set $M_0^l = \{(\emptyset, 0)\}$.
2. For $j = \overline{1_l, n_l}$ do steps a), b) and c):
   a) Set $M_j^l = \emptyset$;
   b) Add each element $(S, c) \in M_{j-1}^l$ to M_j^l. Then for each $(S, c) \in M_{j-1}^l$ form the element $(S \cup \{j_l\}, c + c_{j_l})$ if $\sum\limits_{i\in S} d'_i + d'_{j_l} \le D_l$, $|I_l \cap (S \cup \{j_l\})| \le k_l$, and add $(S \cup \{j_l\}, c + c_{j_l})$ to M_j^l;
   c) Find in M_j^l elements (S, c) and (S', c') with the same second components. For each pair (S, c) and (S', c) we delete (S', c) from M_j^l if $\sum\limits_{i\in S'} d'_i \ge \sum\limits_{i\in S} d'_i$; otherwise delete (S, c).
3. Find in M_n^l the element (S, c) with the maximal second component. Then S is a solution of the packing knapsack problem.

To solve the main problem we shall use the following algorithm.

Algorithm 2

1. For each $l = \overline{1, p}$ solve the corresponding auxiliary problem.

2. Denote by M_n^l the sets obtained for $l = \overline{1,p}$, when the algorithm 1 is applied. Set M_n will contain elements which are obtained as the possible combination of the elements from M_n^l: for each $(S^l, c^l) \in M_n^l$, $l = \overline{1,p}$, element $(S,c) = (\bigcup_l S^l, \sum_l c^l)$ is added to M_n if $\sum_{i \in S} d_i \leq D$.

3. Find in M_n the element (S,c) with the maximal second component. Then S is a solution of the packing knapsack problem.

The algorithm 1 is an extension of the algorithm for a classical knapsack problem from [3]. Therefore the correctness of the algorithm can be argued in analogous way.

Theorem 1. *Algorithm 1 finds the optimal solution of the auxiliary problem in time $O(n_l^2 c^l)$, where n_l is the number of items in the set I_l and $c^l = \sum_{j \in I_l} c_j$.*

To prove this theorem we need the following lemma.

Lemma 1. *Let be $(S,c) \in M_j^l$ after algorithm's running. Then:*
a) $S \subseteq \{1_l, 2_l, \ldots, j_l\}$;
b) $\sum_{i \in S} c_i = c$;
c) $\sum_{i \in S} d'_i \leq D_l$;
d) $|I_l \cap S| \leq k_l$;
e) *if* $(S',c) \in M_j^l$ *then* $S' = S$;
f) *if* $S' \subseteq \{1_l, 2_l, \ldots, j_l\}$ *and* $\sum_{i \in S'} c_i = c$, $|I_l \cap S'| \leq k_l$,
then $\sum_{i \in S} d'_i \leq \sum_{i \in S'} d'_i$;
g) *moreover, if* $S \subseteq \{1_l, 2_l, \ldots, j_l\}$ *and* $\sum_{i \in S} d'_i \leq D_l$, $\sum_{i \in S} c_i = c$, $|I_l \cap S| \leq k_l$, *then there exists* $(S',c) \in M_j^l$;

Proof. We use the induction principle on the iteration number j. The statement for $j = 0$ is evident. Now we consider that $j > 0$ and $(S, c) \in M_j^l$. Here two cases may be:

Case 1. $j \notin S$. Then the element (S, c) has been transferred from M_{j-1}^l in M_j^l at step 2(b). Therefore the properties a), b), c), d), and e) follow from the induction principle.

Case 2. $j \in S$. Then $(S\backslash\{j\}, c - c_j) \in M_{j-1}^l$ and the properties a), b), c), d), and e) hold too.

Now let us prove *f)*. Suppose that $S \neq S'$ and analyze the following three cases:

Case 1. $j \notin S, S'$. Then (S, c) and $(S', c) \in M_{j-1}^l$. So, according to the induction principle $S = S'$.

Case 2. $j \in S, S'$. Then $(S\backslash\{j\}, c - c_j)$, $(S'\backslash\{j\}, c - c_j) \in M_{j-1}^l$. Therefore $S = S'$.

Case 3. $j \in S, j \notin S'$ or $j \notin S, j \in S'$. Then (S', c) was eliminated at the step 2(c). Therefore f) holds.

Now let us prove the property *g)*. We shall use the induction principle on the number $k = maxS$, were $maxS$ is the maximum number of the elements of the subsets S. This property holds for $S = \oslash$. Let us consider $k = maxS > 0$. Then according to the induction principle in M_{k-1}^l there exists element $(S\backslash\{k\}, c - c_k) = (S', c - c_k)$. Therefore the element $(S' \cup \{k\}, c)$ has been added to M_k^l at the step 2(b). Then either $(S' \cup \{k\}, c) \in M_j^l$. So, the property *f)* holds $\square$.

Proof of Theorem 1. The correctness of the algorithm 1 follows from Lemma 1.

Since at the step 3 we find the element (S, c) with the maximal second component and take the first component S as the solution of the packing knapsack problem, the set S is an admissible solution (conform properties c) and d)), its cost is equal to c (conform b)) and a better admissible solution does not exist (conform g)).

Let us find the estimation time of the algorithm 1. Observe, that the power of each set M_{j-1}^l does not exceed c, because two elements with the same second component don't exist in M_{j-1}^l (conform e)). Each modifier operations at the step 2(b) can be implemented in the

time $O(n^l)$, and must be repeated for all $O(c^l)$ elements from M_{j-1}^l. The step 2(c) needs $O(n^l c^l)$ time too, because it may be implemented simultaneously with the step 2(b): for that all elements of M_j^l are memorized in a table with the length equal to c^l, are numbered by second component, and when an other element is tried to be put at a place, the sums of the volumes d_j' are calculated and compared, condition $|I_l \bigcap S| \leq k_l$ is verified and the element, that does not implement this condition or, otherwise, element with the greater volumes sum, is eliminated. Therefore the algorithm finds the optimal solution in time $O(n_l^2 c^l)$ $\square$.

Theorem 2. *Algorithm 2 finds the optimal solution of the packing knapsack problem in time* $O(pn^2c + c^p)$, *where* $c = \sum_{j \in I} c_j$.

Proof . The correctness of the algorithm 2 can be argued in following way.

We obtain the sets $M_n^l, l = \overline{1,p}$, when the algorithm 1 is applied to solve p auxiliary problems. The first components of elements of the sets M_n^l form all admissible solutions for the corresponding auxiliary problem. At the step 2 all possible combination of elements from different sets $M_n^l, l = \overline{1,p}$ are formed if the sum of items volumes from $\bigcup_{l=1}^p S^l$ does not exceed D. Therefore, on the grounds of the theorem 1 and of the condition from step 2, sets M_n contain all admissible solutions of the main problem. Thus, the correctness of the algorithm is argued.

Since at the step 3 we find the element (S, c) with the maximal second component and take the first component S as the solution of the packing knapsack problem, the set S is an admissible solution, its cost is equal to c and a better admissible solution does not exist.

Let us find the estimation time of the algorithm 2. At the step 1 we solve p auxiliary problems, which have the estimation time $O(n^2c)$ each of them. Therefore the step 1 needs time $O(pn^2c)$.

At the step 2 we form all possible combinations of p elements from c elements, therefore we need time $O(c^p)$.

Then, the estimation time of algorithm 2 is $O(pn^2c + c^p)$ $\square$.

References

[1] D. Lozovanu, E. Tataru, A. Zelikovsky. *A generalization of knapsack problem and finding the k-optimal tree in weighted digraph.* Bulletin of Moldova Academy, ser. Math., (1998), pp. 223–239.

[2] E. Tataru. *A generalization of knapsack problem.* Numerical analysis and optimization studies, Moldova State University, V.3, N. 2(6), (2001), pp. 16–19.

[3] Ch.H. Papadimitriu, K.Steiglitz. *Combinatorial optimization: Algorithm and complexity.* New Jersey, 1982.

Elena Balan,

Received December 1, 2004

Institute of Mathematics and Computer Science,
Academy of Sciences, Moldova
str. Academy 5, Chişinău,
2028–MD, Moldova
Phone: (+3732)76 − 05 − 60