Theoretical Computer Science 3 (1976) 293–304.

APPROXIMATE ALGORITHMS FOR SOME GENERALIZED KNAPSACK PROBLEMS

Ashok K. CHANDRA, D.S. HIRSCHBERG[1], C.K. WONG
IBM Thomas J. Watson Research Center, Yorktown Heights, NY 10598, U.S.A.

Communicated by Richard Karp
Received 5 January 1976
Revised 14 April 1976

Abstract. In this paper we construct approximate algorithms for the following problems: integer multiple-choice knapsack problem, binary multiple-choice knapsack problem and multi-dimensional knapsack problem. The main result can be described as follows: for every $\varepsilon > 0$ one can construct a polynomial-time algorithm for each of the above problems such that the ratio of the value of the objective function by this algorithm and the optimal value is bounded below by $1 - \varepsilon$.

1. Introduction

In a recent study of the problem of secondary index selection for a large data base in a multi-level storage, the following optimization problem arose [8, p. 319]:

Given a_{ij}, b, c_{ij}, with $0 \leqslant c_{ij}$, $0 \leqslant a_{ij}$, $i = 1, 2, \ldots, n$, $j = 1, 2, \ldots, k$, find a binary n-vector $x = (x_{1,j_1}, \ldots, x_{n,j_n})$, (i.e. x_{i,j_i} is 0 or 1 only) such that $\sum_{i=1}^{n} a_{i,j_i} x_{i,j_i} \leqslant b$ and $\sum_{i=1}^{n} c_{i,j_i} x_{i,j_i}$ is maximized.

This is a generalized knapsack problem. (For details of the knapsack problem, see [1], for example). In [8], it is solved by dynamic programming methods. But the computation time becomes prohibitively large as the size of the problem grows. In fact, it has been shown that both the integer knapsack problem and the binary knapsack problem are NP-complete. (See, for example, [4, 6].) Consequently, the present generalized knapsack problem is also NP-complete. In view of this, it is necessary to find good heuristics. The approach which we adopt in this paper was first discovered by Johnson [3] and has been successfully applied by various authors. (See, for example, [2, 7].)

However, for this approach to work, one has first to find a heuristic which yields results within some suitable additive constant of the optimum. Such a heuristic is

[1] Current address: Rice University, Department of Electrical Engineering, Houston, Texas 77001, U.S.A.

proposed in the paper. To show that it satisfies the requirement, we resort to a "continuous" argument.

In addition to the above-mentioned problem, which will be referred to as the binary multiple-choice knapsack problem, we also consider the integer multiple-choice knapsack problem, i.e., the components of the solution vector $\boldsymbol{x} = (x_{1,j_1}, \ldots, x_{n,j_n})$ in the above problem can be any non-negative integers. Finally, we also study the integer multi-dimensional knapsack problem:

$$\text{maximize} \quad \sum_{i=1}^{n} c_i x_i$$

$$\text{subject to} \quad \sum_{i=1}^{n} a_{i1} x_i \leqslant b_1,$$

$$\vdots$$

$$\sum_{i=1}^{n} a_{ik} x_i \leqslant b_k,$$

where the solution $\{x_i\}$ are non-negative integers.

We shall solve the integer multiple-choice knapsack problem first since it is much simpler and demonstrates the general approach better. The solution to the binary version of the multi-dimensional knapsack problem is still not known. The results obtained in this paper can be briefly described as follows: for every $\varepsilon > 0$, we can find a polynomial-time algorithm (ε-algorithm) such that for each input for which the optimum value of the objective function $C_0 \neq 0, \infty$, $C_A / C_0 \geqslant 1 - \varepsilon$, where C_A is the value of the objective function by the algorithm.

It should be pointed out that throughout this paper the algorithms are presented in their simplest forms for the sake of clarity. They can be modified to take advantage of various programming techniques to yield slightly better results.

2. The integer multiple-choice knapsack problem

The problem is: Given positive integers n, k, and non-negative rationals b, c_{ij}, a_{ij}, $i = 1, \ldots, n$, $j = 1, \ldots, k$, find n-vectors of integers $\boldsymbol{x} = (x_1, \ldots, x_n)$, $j = (j_1, \ldots, j_n)$, $0 \leqslant x_i$, $1 \leqslant j_i \leqslant k$, such that

(i) $\sum_{i=1}^{n} a_{i,j_i} x_i \leqslant b$,

(ii) $\sum_{i=1}^{n} c_{i,j_i} x_i$ is maximized.

The problem may be reformulated as follows: Given positive integers n, k, and non-negative rationals b, a_{ij}, c_{ij}, find an nk-vector of integers $(x_{11}, x_{12}, \ldots, x_{nk})$, $0 \leqslant x_{ij}$, subject to

(i) $\sum_{i=1}^{n} \sum_{j=1}^{k} a_{ij} x_{ij} \leqslant b$,

(ii) for each $i = 1, \ldots, n$ there is at most one j for which $x_{ij} > 0$,

(iii) $\sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}$ is maximized.

We first consider the following algorithm:

Algorithm IM. If all $a_{ij} > b$, or all $c_{ij} = 0$, the solution is $\boldsymbol{x} = (0, \ldots, 0)$ and if for any i, j, $a_{ij} = 0$ and $c_{ij} > 0$ then the objective function is unbounded. Otherwise, let $\rho_{ij} = c_{ij}/a_{ij}$ for $1 \leq i \leq n$, $1 \leq j \leq k$, (excluding those i, j for which $c_{ij} = a_{ij} = 0$) and let $\rho_{tf} = \text{Max}\{\rho_{ij} \mid 0 < a_{ij} \leq b\}$. As a solution take $x_{tf} = \lfloor b/a_{tf} \rfloor$, and other $x_{ij} = 0$. The value of the objective function is $c_{tf} \lfloor b/a_{tf} \rfloor$.

Next, we construct an ε-algorithm:

Algorithm IM$_\varepsilon$. If, for any i, j, $a_{ij} = 0$ and $c_{ij} > 0$ then the objective function is unbounded. Otherwise, let $\delta = b\varepsilon/(1+\varepsilon)$. Parition the coefficients $\{a_{ij}\}$ into two parts: those larger than or equal to δ, and those less than δ. Without loss of generality, we assume that $a_{ij} \geq \delta$ for $i = 1, \ldots, n$, $j = 1, \ldots, p_i$ $(0 \leq p_i \leq k)$, and $a_{ij} < \delta$ otherwise. Let Ω be the set of all nk-vectors $\boldsymbol{x} = (x_{11}, \ldots, x_{nk})$ of integers $x_{ij} \geq 0$, $x_{ij} = 0$ for $j > p_i$, for each i at most one $x_{ij} > 0$, and $\sum\sum a_{ij}x_{ij} \leq b$. For each $\boldsymbol{x} \in \Omega$, apply algorithm IM to the following problem $P(\boldsymbol{x})$:

Find an nk-vector $\boldsymbol{x}' = (x'_{11}, \ldots, x'_{nk})$ such that

(i) $\sum\sum a'_{ij}x'_{ij} \leq b - \sum\sum a_{ij}x_{ij}$,

(ii) For each $i = 1, \ldots, n$ there is at most one j for which $x'_{ij} > 0$,

(iii) $\sum\sum c_{ij}x'_{ij}$ is maximized,

where

$$a'_{ij} = \begin{cases} b+1 & \text{if } j \leq p_i, \text{ or for some } s, x_{is} > 0, \\ a_{ij} & \text{otherwise.} \end{cases}$$

Note: if some $a'_{ij} = b + 1$, that effectively enforces $x'_{ij} = 0$. Let $C_{\text{IM}}(\boldsymbol{x})$ be the value obtained from IM, and let $C(\boldsymbol{x}) = \sum\sum c_{ij}x_{ij}$. Then, algorithm IM$_\varepsilon$ chooses that solution $\boldsymbol{x} + \boldsymbol{x}'$ which maximizes $C(\boldsymbol{x}) + C_{\text{IM}}(\boldsymbol{x})$, i.e., the objective function has value

$$C_{\text{IM}_\varepsilon} = \max_{\boldsymbol{x} \in \Omega} (C(\boldsymbol{x}) + C_{\text{IM}}(\boldsymbol{x})).$$

Theorem 2.1. (i) *Algorithm IM$_\varepsilon$ can be implemented in* $O((kn)^{\lfloor 1/\varepsilon \rfloor + 1})$ *time and* $O(n)$ *space.*

(ii) *If* $C_0 \neq 0, \infty$, *then* $C_{\text{IM}_\varepsilon}/C_0 > 1 - \varepsilon$.

Proof. (i) $|\Omega| = O((kn)^{\lfloor 1/\varepsilon \rfloor + 1})$, the elements can be enumerated in this much time, and for each $x \in \Omega$, $C_{\text{IM}}(x)$ can be computed in time $O(1)$. To do this, we precompute for each i, that value j_i which is the value of j that maximizes $\rho_{ij} = c_{ij}/a_{ij}$ for $j > p_i$. These n values are sorted in descending order of ρ_{i,j_i}, and we may assume that $\rho_{1,j_1} \geq \rho_{2,j_2} \geq \cdots$. As the elements $x \in \Omega$ are enumerated, one keeps track of $b - \sum\sum a_{ij}x_{ij}$, and the minimum i for which $x_{ij} = 0$ for all j. Space required is $O(n)$.

(ii) Let $\boldsymbol{y}$ be an optimal solution. Let $\boldsymbol{x} = (x_{11}, \ldots, x_{nk})$, where $x_{ij} = y_{ij}$ if $j \leq p_i$,

and $x_{ij} = 0$ otherwise. Then $x \in \Omega$. Consider the application of algorithm IM to problem $P(x)$, having value $C_{IM}(x)$. If c_{ij}/a_{ij} is the largest among $\{c_{ij}/a_{ij} \mid 0 < a_{ij} \leq b_1\}$ where $b_1 = b - \sum\sum a_{ij}x_{ij}$, the optimal value $C_0(x)$ for problem $P(x)$ is bounded above by $c_{ij}b_1/a_{ij}$, hence

$$C_0 - C_{IM_\varepsilon} \leq (C(x) + C_0(x)) - (C(x) + C_{IM}(x)) < c_{ij}.$$

On the other hand, for problem $P(x_0)$ where $x_0 = (0, \ldots, 0)$, let c_{ij}/a_{ij} maximize $\{c_{ij}/a_{ij} \mid 0 < a_{ij} \leq b\}$. Then

$$C_{IM}(x_0) = c_{ij}\left[\frac{b}{a_{ij}}\right] > c_{ij}\frac{b-\delta}{a_{ij}} \geq c_{ij}\frac{b-\delta}{a_{ij}} > c_{ij}\frac{b-\delta}{\delta} = \frac{c_{ij}}{\varepsilon}$$

(as $a_{ij} < \delta$). Thus

$$\frac{C_{IM_\varepsilon}}{C_0} = 1 - \frac{C_0 - C_{IM_\varepsilon}}{C_0} > 1 - \frac{c_{ij}}{(c_{ij})/\varepsilon} = 1 - \varepsilon. \quad \square$$

Remark. For $\varepsilon \geq \frac{1}{2}$, algorithm IM_ε could be improved to run in time $O(kn)$ by making it identical with algorithm IM.

3. The binary multiple-choice knapsack problem

The problem is: Given positive integers n, k, and non-negative rationals b, a_{ij}, c_{ij}, $i = 1, \ldots, n$, $j = 1, \ldots, k$, find n-vectors of integers $x = (x_1, \ldots, x_n)$ and $j = (j_1, \ldots, j_n)$, $0 \leq x_i \leq 1$, $0 \leq j_i \leq k$, such that

(i) $\sum_{i=1}^{n} a_{i,j_i}x_i \leq b$,

(ii) $\sum_{i=1}^{n} c_{i,j_i}x_i$ is maximized.

The problem may be reformulated as follows: Given positive integers, n, k, and non-negative rationals b, a_{ij}, c_{ij}, find an nk-vector of integers $(x_{11}, x_{12}, \ldots, x_{nk})$, $0 \leq x_{ij} \leq 1$, subject to

(i) $\sum_{i=1}^{n}\sum_{j=1}^{k} a_{ij}x_{ij} \leq b$,

(ii) $\sum_{j=1}^{k} x_{ij} \leq 1$ for $i = 1, \ldots, n$, (1)

(iii) $\sum_{i=1}^{n}\sum_{j=1}^{k} c_{ij}x_{ij}$ is maximized.

Let R be the "continuous" version of the above problem, the only change being that x_{ij} can be real numbers ($0 \leq x_{ij} \leq 1$), not just integers. In the sequel we will assume that there is no i, j for which $a_{ij} > b$ (without loss of generality), or $a_{ij} = 0$ and $c_{ij} > 0$ (for the solution is trivial).

Lemma 3.1. *For problem R, if there is an integer s, and distinct p, q, such that $a_{sp}, a_{sq}, c_{sp} > 0$, $c_{sp}/a_{sp} \geq c_{sq}/a_{sq}$, and $c_{sp} \geq c_{sq}$, then if $x = (x_{11}, \ldots, x_{nk})$ is any solution, there is another solution $x' = (x'_{11}, \ldots, x'_{nk})$ for which $x'_{sq} = 0$, and for all i, j, if $x_{ij} = 0$ then $x'_{ij} = 0$ except x'_{sp} (if $a_{sp} = a_{sq} = 0$ and $c_{sp} > c_{sq}$, we consider $c_{sp}/a_{sp} > c_{sq}/a_{sq}$).*

Proof. Choose

$$x'_{sp} = x_{sp} + \frac{c_{sq}}{c_{sp}} x_{sq},$$

$$x'_{sq} = 0,$$

$$x'_{ij} = x_{ij} \quad \text{for } i \neq s \text{ or } j \neq p, q.$$

Then

$$\sum_{i=1}^{n} \sum_{j=1}^{k} a_{ij} x'_{ij} = \sum_{i=1}^{n} \sum_{j=1}^{k} a_{ij} x_{ij} - a_{sp} x_{sp} - a_{sq} x_{sq} + a_{sp} \left(x_{sp} + \frac{c_{sq}}{c_{sp}} x_{sq} \right)$$

$$\leqslant b - \frac{x_{sq}}{c_{sp}} (c_{sp} a_{sq} - c_{sq} a_{sp}) \leqslant b,$$

and for all i,

$$\sum_{j=1}^{k} x'_{ij} \leqslant \sum_{j=1}^{k} x_{ij} \leqslant 1 \text{ as } c_{sq} \leqslant c_{sp},$$

and

$$\sum_{i=1}^{n} \sum_{j=1}^{k} c_{ij} x'_{ij} = \sum_{i=1}^{n} \sum_{j=1}^{k} c_{ij} x_{ij}. \quad \square$$

Lemma 3.2. *For problem R, if there is an integer s, and distinct p, q, r such that $c_{sp} \leqslant c_{sq} \leqslant c_{sr}$, $a_{sp} \leqslant a_{sq} \leqslant a_{sr}$, and either $c_{sq} = c_{sp}$ or $a_{sr} = a_{sq}$ or $(c_{sq} - c_{sp})/(a_{sq} - a_{sp}) \leqslant (c_{sr} - c_{sq})/(a_{sr} - a_{sq})$, then if $\boldsymbol{x} = (x_{11}, \ldots, x_{nk})$ is any solution, then there is another solution $\boldsymbol{x}' = (x'_{11}, \ldots, x'_{nk})$ for which $x'_{sq} = 0$, and for all i, j, if $x_{ij} = 0$ then $x'_{ij} = 0$ except x'_{sp} and x'_{sr}.*

Proof. If $c_{sq} \neq c_{sp}$ and $a_{sr} \neq a_{sq}$ then choose

$$x'_{sp} = x_{sp} + x_{sq} \frac{c_{sr} - c_{sq}}{c_{sr} - c_{sp}},$$

$$x'_{sq} = 0,$$

$$x'_{sr} = x_{sr} + x_{sq} \frac{c_{sq} - c_{sp}}{c_{sr} - c_{sp}},$$

$$x'_{ij} = x_{ij} \quad \text{for } i \neq s \text{ or } j \neq p, q, r.$$

Then

$$\sum_{i=1}^{n} \sum_{j=1}^{k} a_{ij} x'_{ij} = \sum_{i=1}^{n} \sum_{j=1}^{k} a_{ij} x_{ij} + x_{sq} \left(a_{sp} \frac{c_{sr} - c_{sq}}{c_{sr} - c_{sp}} - a_{sq} + a_{sr} \frac{c_{sq} - c_{sp}}{c_{sr} - c_{sp}} \right)$$

$$\leqslant b + \frac{x_{sq}}{c_{sr} - c_{sp}} (-(c_{sr} - c_{sq})(a_{sq} - a_{sp}) + (c_{sq} - c_{sp})(a_{sr} - a_{sq}))$$

$$\leqslant b$$

and for all i,

$$\sum_{j=1}^{k} x'_{ij} = \sum_{j=1}^{k} x_{ij} \leq 1,$$

and

$$\sum_{i=1}^{n} \sum_{j=1}^{k} c_{ij} x'_{ij} = \sum_{i=1}^{n} \sum_{j=1}^{k} c_{ij} x_{ij}.$$

On the other hand, if $c_{sq} = c_{sp}$ then choose $x'_{sp} = x_{sp} + x_{sq}$, $x'_{sq} = 0$, $x'_{sr} = x_{sr}$, and if $a_{sr} = a_{sq}$, then choose $x'_{sp} = x_{sp}$, $x'_{sq} = 0$, $x'_{sr} = x_{sr} + x_{sq}$. In both cases the lemma is easily verified. □

Given problem R, we can eliminate variables using Lemmas 3.1, 3.2 such that we have the reduced problem R':

(i) $\sum_{i=1}^{n} \sum_{j=1}^{q_i} a_{ij} x_{ij} \leq b$,

(ii) $\sum_{j=1}^{q_i} x_{ij} \leq 1$ for $i = 1, \ldots, n$,

(iii) $\sum_{i=1}^{n} \sum_{j=1}^{q_i} c_{ij} x_{ij}$ is maximized,

where for each i:

$$0 < c_{i1} < c_{i2} < \cdots < c_{i,q_i}$$

$$\frac{c_{i1}}{a_{i1}} > \frac{c_{i2}}{a_{i2}} > \cdots > \frac{c_{i,q_i}}{a_{i,q_i}},$$

i.e. $a_{i1} < a_{i2} < \cdots < a_{i,q_i}$ and

$$\frac{c_{i1}}{a_{i1}} > \frac{c_{i2} - c_{i1}}{a_{i2} - a_{i1}} > \cdots > \frac{c_{i,q_i} - c_{i,q_i-1}}{a_{i,q_i} - a_{i,q_i-1}}.$$

We now transform R' into the "easier" knapsack-like problem R'' below, whose objective function has value at least as large as that of R'. As it will turn out, the two are equal. R'' is:

Given $n \geq 1$, $q_i \geq 0$ for $i = 1, \ldots, n$, positive rationals a'_{ij}, c'_{ij}, b, find a vector of real numbers $y = (y_{11}, \ldots, y_{n,q_n})$ such that

(i) $\sum_{i=1}^{n} \sum_{j=1}^{q_i} a'_{ij} y_{ij} \leq b$,

(ii) For each i, j $0 \leq y_{ij} \leq 1$,

(iii) $\sum_{i=1}^{n} \sum_{j=1}^{q_i} c'_{ij} y_{ij}$ is maximized.

For a given problem R', the corresponding problem R'' has

$$a'_{ij} = \begin{cases} a_{ij} & \text{for } j = 1, \\ a_{ij} - a_{i,j-1} & \text{for } j > 1, \end{cases}$$

$$c'_{ij} = \begin{cases} c_{ij} & \text{for } j = 1, \\ c_{ij} - c_{i,j-1} & \text{for } j > 1. \end{cases}$$

Lemma 3.3. *The maximum value of the objective function for problem R' is the same as that for the corresponding problem R''.*

Proof. we first show that the objective function for R'' can be at least as large as that for R'. Given a solution $(x_{11}, \ldots, x_{n,q_n})$ of R', we have a corresponding $y = (y_{11}, \ldots, y_{n,q_n})$ where

$$y_{ij} = \sum_{s=j}^{q_i} x_{is}.$$

Then

$$\begin{aligned}\sum_{i=1}^{n}\sum_{j=1}^{q_i} a'_{ij}y_{ij} &= \sum_{i=1}^{n}\sum_{j=1}^{q_i}\sum_{s=j}^{q_i} a'_{ij}x_{is} \\ &= \sum_{i=1}^{n}\sum_{s=1}^{q_i}\sum_{j=1}^{s} a'_{ij}x_{is} \\ &= \sum_{i=1}^{n}\sum_{s=1}^{q_i} x_{is}(a_{i1} + (a_{i2} - a_{i1}) + \cdots + (a_{is} - a_{i,s-1})) \\ &= \sum_{i=1}^{n}\sum_{s=1}^{q_i} x_{is}a_{is} \leqslant b\end{aligned}$$

and

$$y_{ij} \leqslant \sum_{s=1}^{q_i} x_{is} \leqslant 1$$

and

$$\sum_{i=1}^{n}\sum_{j=1}^{q_i} c'_{ij}y_{ij} = \sum_{i=1}^{n}\sum_{s=1}^{q_i} x_{is}c_{is}$$

(like the above) i.e., the value of the objective function is the same.

The solution of problem R'' is easy. Simply order the multiset $\{c'_{ij}/a'_{ij}\}$ such that $c'_{i_0,j_0}/a'_{i_0,j_0} \geqslant c'_{i_1,j_1}/a'_{i_1,j_1} \geqslant \cdots$, then let r be the integer such that $\sum_{p=0}^{r-1} a'_{i_p,j_p} \leqslant b < \sum_{p=0}^{r} a'_{i_p,j_p}$ (assuming $\sum_{i=1}^{n}\sum_{j=1}^{q_i} a'_{ij} > b$, for otherwise the problem is trivial: choose all $y_{i,j} = 1$), then

$$y_{i_p,j_p} = \begin{cases} 1 & \text{for } p \leqslant r-1 \\ \alpha & \text{for } p = r \\ 0 & \text{for } p > r, \end{cases}$$

where

$$\alpha = \left(b - \sum_{q=1}^{r-1} a'_{i_q,j_q}\right) \Big/ a'_{i_r,j_r} < 1.$$

If, however, there is an R' to which this problem R'' corresponds, this also yields a solution of the problem R' below, having the same value of the objective function:

$x_{i_r,j_r} = \alpha$

$x_{i_r,j_{r-1}} = 1 - \alpha$ (unless $j_r = 1$ in which case this is meaningless),

$x_{i_r,j} = 0$ for $j \neq j_{r-1}, j_r$,

$x_{i,j'_i} = 1$ for $i \neq i_r$ (j'_i defined below),

$x_{ij} = 0$ otherwise,

where for each i, j'_i = the maximum j for which the pair (i, j) appears in the sequence $(i_0, j_0), \ldots, (i_{r-1}, j_{r-1})$, and $j'_i = 0$ if there is no such j.

Then

$$\begin{aligned}
\sum_{i=1}^{n}\sum_{s=1}^{q_i} a_{is}x_{is} &= \sum_{\substack{1\le i\le n\\ i\ne i_r}}\sum_{1\le s\le q_i} a_{is}x_{is} + \sum_{1\le s\le q_{i_r}} a_{i_r,s}x_{i_r,s} \\
&= \sum_{\substack{1\le i\le n\\ i\ne i_r}} a_{ij'_i} + a_{i_r,j_{r-1}}(1-\alpha) + a_{i_r,j_r}\alpha \\
&= \sum_{\substack{1\le i\le n\\ i\ne i_r}} a_{i1} + (a_{i2}-a_{i1}) + \cdots + (a_{i,j'_i} - a_{i,j'_i-1}) \\
&\quad + a_{i_r,1} + (a_{i_r,2} - a_{i_r,1}) + \cdots + (a_{i_r,j_r-1} - a_{i_r,j_r-2}) \\
&\quad + \alpha(a_{i_r,j_r} - a_{i_r,j_r-1}) \\
&= \sum_{\substack{1\le i\le n\\ i\ne i_r}}\sum_{1\le s\le q_i} a'_{is}y'_{is} + \sum_{1\le s\le q_{i_r}} a'_{i_r,s}y'_{i_r,s} \\
&= b.
\end{aligned}$$

And similarly

$$\sum_{i=1}^{n}\sum_{s=1}^{q_i} c_{is}x_{is} = \sum_{i=1}^{n}\sum_{s=1}^{q_i} c'_{is}y'_{is}. \quad \square$$

We now return to the integer version of problem R, i.e. the problem (1), and consider the following algorithm.

Algorithm BM. Convert the problem to R'' and find the integer r as above. Then choose, for all i, j

$$x_{ij} = \begin{cases} 1 & \text{if } j = j'_i, \\ 0 & \text{otherwise.} \end{cases}$$

(Note: j'_i is defined above and $j'_{i_r} = j_r - 1$ by definition.) Compare $\sum_{i=1}^{n}\sum_j c_{ij}x_{ij}$ with c_{i_r,j_r} and if the former is at least as large as the latter, the algorithm returns $\{x_{ij}\}$ as solution, otherwise it returns as solution $x_{i_r,j_r} = 1$ and all other $x_{ij} = 0$.

Let C_{BM} be the value of this algorithm. The optimum value C_0 for the continuous version of this algorithm is, from the solution above,

$$\begin{aligned}
C_0 &= \sum_{i=1}^{n} c_{i,j'_i} + \alpha(c_{i_r,j_r} - c_{i_r,j_r-1}) \\
&< \sum_{i=1}^{n} c_{i,j'_i} + c_{i_r,j_r} \\
&\le 2C_{BM}. \qquad (2)
\end{aligned}$$

Also

$$C_0 - C_{\mathrm{BM}} < c_{i_r j_r}. \tag{3}$$

We now construct an ε-algorithm:

Algorithm BM$_\varepsilon$. Let $\delta = \lceil (1/\varepsilon) - 2 \rceil$, and let Ω be the set of all binary nk vectors $\boldsymbol{x} = (x_{11}, \ldots, x_{nk})$ with at most δ components being 1, and such that for each $i = 1, \ldots, n$, $\sum_{j=1}^{k} x_{ij} \leqslant 1$ and $\sum_{i=1}^{n} \sum_{j=1}^{k} a_{ij} x_{ij} \leqslant b$. For each $\boldsymbol{x} \in \Omega$, let $\hat{c} = \min\{c_{ij} \mid x_{ij} = 1\}$, and apply algorithm BM to the following problem $T_{\boldsymbol{x}}$:

$$\begin{aligned} \text{maximize} \quad & \sum_{i=1}^{n} \sum_{j=1}^{k} c'_{ij} x'_{ij} \\ \text{subject to} \quad & \sum_{i=1}^{n} \sum_{j=1}^{k} a_{ij} x'_{ij} \leqslant b - \sum_{i=1}^{n} \sum_{j=1}^{k} a_{ij} x_{ij}, \\ & \sum_{j=1}^{k} x'_{ij} \leqslant 1 \quad \text{for } i = 1, \ldots, n, \\ & x'_{ij} \in \{0, 1\} \quad \text{for } i = 1, \ldots, n,\ j = 1, \ldots, k, \end{aligned}$$

where

$$c'_{ij} = \begin{cases} 0 & \text{if } x_{il} = 1 \text{ for any } l, \text{ or if } c_{ij} > \hat{c}, \\ c_{ij} & \text{otherwise.} \end{cases}$$

If $\boldsymbol{x}'$ is the output, then $\boldsymbol{x} + \boldsymbol{x}'$ is a feasible solution of the original problem since $x'_{ij} = 0$ whenever $c'_{ij} = 0$. Let $C(\boldsymbol{x}) = \sum \sum c_{ij} x_{ij}$, and $C_{\mathrm{BM}}(\boldsymbol{x}) = \sum \sum c'_{ij} x'_{ij}$. The algorithm BM$_\varepsilon$ chooses that solution which has value

$$C_{\mathrm{BM}_\varepsilon} = \max_{\boldsymbol{x} \in \Omega} (C(\boldsymbol{x}), C_{\mathrm{BM}}(\boldsymbol{x})).$$

Theorem 3.4. (i) *Algorithm BM$_\varepsilon$ takes time* $O((nk)^{\lceil (1/\varepsilon) - 1 \rceil} \log n)$ *and space* $O(nk)$.
(ii) *If* $C_0 \neq 0, \infty$, *then* $C_{\mathrm{BM}_\varepsilon} / C_0 > 1 - \varepsilon$.

Proof. (i) $|\Omega| \leqslant (nk)^{\delta} = (nk)^{\lceil (1/\varepsilon) - 2 \rceil}$, and for each $\boldsymbol{x} \in |\Omega|$, $C(\boldsymbol{x})$, $C_{\mathrm{BM}}(\boldsymbol{x})$ can be computed in time $O(nk \log n)$.

(ii) Let $\boldsymbol{y}$ be an optimal solution having value $C_0 = \sum \sum c_{ij} y_{ij}$. If the number of nonzero elements of $\boldsymbol{y} \leqslant \delta$, BM$_\varepsilon$ produces an optimal solution. Otherwise let $\boldsymbol{x} = (x_{11}, \ldots, x_{nk})$ be defined to be the same as $\boldsymbol{y}$ for the δ largest values y_{ij}, and 0 otherwise. Then $\boldsymbol{x} \in \Omega$, and

$$\frac{C_{\mathrm{BM}_\varepsilon}}{C_0} \geqslant \frac{C(\boldsymbol{x}) + C_{\mathrm{BM}}(\boldsymbol{x})}{\sum \sum c_{ij} y_{ij}} = 1 - \frac{\sum \sum c_{ij} (y_{ij} - x_{ij}) - c_{\mathrm{BM}}(\boldsymbol{x})}{\sum \sum c_{ij} (y_{ij} - x_{ij}) + C(\boldsymbol{x})}.$$

But, from (3) above,

$$\sum \sum c_{ij}(y_{ij} - x_{ij}) - C_{BM}(\boldsymbol{x}) < c_{ir,jr} \leqslant \hat{c} \leqslant \frac{1}{\delta} C(\boldsymbol{x}),$$

(note: $\hat{c}$ defined in the description of BM_ε), and since from (2)

$$\sum \sum c_{ij}(y_{ij} - x_{ij}) < 2C_{BM}(\boldsymbol{x}),$$

$$\sum \sum c_{ij}(y_{ij} - x_{ij}) - C_{BM}(\boldsymbol{x}) < \tfrac{1}{2} \sum \sum c_{ij}(y_{ij} - x_{ij}),$$

we have

$$\frac{C_{BM_\varepsilon}}{C_0} > 1 - \frac{1}{\delta + 2} \geqslant 1 - \varepsilon. \quad \square$$

4. The integer Multi-Dimensional Knapsack Problem

For a fixed positive integer k, the k-dimensional knapsack problem is the following:

$$\text{maximize} \quad \sum_{i=1}^{n} c_i x_i$$

$$\text{subject to} \quad \sum_{i=1}^{n} a_{ij} x_i \leqslant b_j, \quad j = 1, \ldots, k,$$

where n is a given positive integer, and c_i, b_i, a_{ij} are given non-negative rationals; the solution is to be in non-negative integers. Without loss of generality, we assume that for each i there is a j for which $a_{ij} > 0$.

If we relax the constraints on x_i to allow non-negative real numbers (this problem will be referred to as S), it is well-known in linear programming that the new problem admits an optimal solution where at most k of x_i's are non-zero. (See e.g. [5], Sec. 12–4.) Therefore we have only to examine all k-combinations of $(1, 2, \ldots, n)$ and for each k-combination, solve a problem of the following from:

$$\text{maximize} \quad \sum_{i=1}^{k} d_i y_i$$

$$\text{subject to} \quad \sum_{j=1}^{k} e_{ij} y_i \leqslant b_j, \qquad j = 1, \ldots, k,$$

which can be solved in time $2^{O(k)}$ (using the big-oh notation). Therefore the whole process takes time $O(n^k)$. (Note that k is a constant.)

We have the following algorithm.

Algorithm MD. Let $x_{i_1}, \ldots, x_{i_k}$ be an optimal solution of S. Take $\lfloor x_{i_1} \rfloor, \ldots, \lfloor x_{i_k} \rfloor$ as an approximate solution for the original problem.

Let C_S be the optimal cost of problem S, let C_{MD} be the value of algorithm MD

and let C_0 be the optimal value of the original multidimensional knapsack problem. Then

$$C_0 - C_{MD} \leqslant C_S - C_{MD} \leqslant c_{i_1} + \cdots + c_{i_k} \leqslant k\hat{c}, \quad \text{where } \hat{c} = \max\{c_1, \ldots, c_n\}.$$

We now construct an ε-algorithm,

Algorithm MD$_\varepsilon$. Order the x_i's such that $c_1 \geqslant c_2 \geqslant \cdots \geqslant c_n$. Let $\delta = \lceil k((1/\varepsilon)-1)\rceil$. Let Ω be the set of all vectors $x = (x_1, \ldots, x_n)$ where each x_i is a non-negative integer, $\sum_{i=1}^n x_i \leqslant \delta$, and $\sum_{i=1}^n a_{ij}x_i \leqslant b_j$, $j = 1, \ldots, k$. For every $x \in \Omega$, apply algorithm MD to the following problem:

Let m be the maximum integer i for which $x_i \neq 0$ (if all $x_i = 0$ choose $m = 0$),

$$\text{maximize} \quad \sum_{i=m+1}^{n} c_i z_i \tag{4}$$

$$\text{subject to} \quad \sum_{i=m+1}^{n} a_{ij} z_i \leqslant b_j - \sum_{i=1}^{n} a_{ij}x_i, \qquad j = 1, \ldots, k.$$

Let $C_{MD}(x)$ be its value, and let $C(x) = \sum_{i=1}^n c_i x_i$. Then algorithm MD$_\varepsilon$ chooses as solution the vector that achieves

$$C_{MD_\varepsilon} = \max_{x \in \Omega} (C(x) + C_{MD}(x)).$$

Theorem 4.1. (i) *Algorithm MD$_\varepsilon$ takes time* $O(n^{\lceil k/\varepsilon \rceil})$ *and* $O(n)$ *space.*

(ii) *If* $C_0 \neq 0, \infty$ *then* $C_{MD_\varepsilon}/C_0 > 1 - \varepsilon$ (*where* C_0 *is the optimum value*).

Proof. (i) $|\Omega| = O(n^\delta)$, and for each $x \in \Omega$, $C_{MD}(x)$ can be computed in $O(n^k)$, thus total time is $O(n^{\lceil k/\varepsilon \rceil})$.

(ii) Suppose C_0 is attained by a vector $y = (y_1, \ldots, y_n)$. If $\sum_{i=1}^n y_i \leqslant \delta$, then $C_{MD_\varepsilon} = C_0$. Otherwise let $x = (y_1, \ldots, y_{m-1}, x_m, 0, 0, \ldots, 0)$ where $y_1 + \cdots + y_{m-1} + x_m = \delta$, $x_m \neq 0$. Then $x \in \Omega$, and for this x, $C_0(x) - C_{MD}(x) < kc_m$ where $C_0(x)$ is the optimum value for problem (4) above. Then

$$\frac{C_{MD_\varepsilon}}{C_0} \geqslant \frac{C(x) + C_{MD}(x)}{C(x) + C_0(x)} \geqslant 1 - \frac{C_0(x) - C_{MD}(x)}{C(x) + C_0(x) - C_{MD}(x)}.$$

But $C(x) \geqslant \delta c_m$ and $C_0(x) - C_{MD}(x) < kc_m$ so that

$$\frac{C_{MD_\varepsilon}}{C_0} > 1 - \frac{kc_m}{\delta c_m + kc_m} \geqslant 1 - \varepsilon$$

(since $x/(a+x)$, $a, x > 0$, is maximized when x is maximized). □

Note added in proof

It has been pointed out by David S. Johnson that the algorithms IM_ε and BM_ε can be improved using the ideas of Ibarra and Kim [2], so as to run in time $O(kP(n)/\varepsilon)$ where $P(n)$ is a low-order polynomial in n. However, we are unaware of any such improvement for the algorithm MD_ε.

References

[1] R.S. Garfinkel and G.L. Nemhauser, *Integer Programming* (John Wiley and Sons, New York, 1972).
[2] O.H. Ibarra and C.E. Kim, Fast approximation algorithms for the knapsack and sum of subset problems *J. Assoc. Comput. Mach.* **22** (1975) 463–468.
[3] D.S. Johnson, Approximation algorithms for combinatorial problems, *J. Comput. System Sci.* **9** (1974) 256–278.
[4] R.M. Karp, Reducibility among combinatorial problems in: R.E. Miller, J.W. Thatcher (Eds.), *Complexity of Computer Computations* (Plenum Press, N.Y., 1972) 85–104.
[5] C.L. Liu, *Introduction to Combinatorial Mathematics* (McGraw-Hill, N.Y., 1968).
[6] G.S. Lueker, Two polynomial complete problems in non-negative integer programming, Computer Science Report TR–178, Princeton University (March 1975).
[7] S. Sahni, Approximate Algorithms for the 0/1 Knapsack Problem, *J. Assoc. Comput. Mach.* **22** (1975) 115–124.
[8] P.C. Yue and C.K. Wong, Storage cost considerations in secondary index selection, *Int. J. Comp. Inf. Sci.* **4** (1975) 307–327.