APPROXIMATE ALGORITHMS FOR THE MULTIPLE-CHOICE CONTINUOUS KNAPSACK PROBLEMS

Toshihide Ibaraki
Kyoto University

(Received November 2, 1978)

Abstract

The multiple-choice continuous knapsack problem is defined as follows: maximize \( z = \sum_{i=1}^{n} \sum_{j=1}^{m_i} a_{ij} x_{ij} \)
subject to \( \sum_{i=1}^{n} \sum_{j=1}^{m_i} a_{ij} x_{ij} \leq b \), \( 0 \leq x_{ij} \leq 1 \), \( i = 1, 2, \ldots, n \), \( j = 1, 2, \ldots, m_i \), (3) at most one of \( x_{ij} (j = 1, 2, \ldots, m_i) \) is positive for each \( i = 1, 2, \ldots, n \), where \( n, m_i \) are positive integers and \( a_{ij} \) are nonnegative integers. In this paper, it is proved that this problem is \( \text{NP-complete} \) (which strongly suggests the computational intractability to obtain exact optimal solutions). Then, starting with an approximate solution obtained from the \( \text{LP} \) optimal solution by rounding, two approximate algorithms are proposed and analyzed. The first one (called the breadth-1 search method) obtains an approximate solution with the (worst case) relative error less than 25%. The required computation time is \( O(mN \log N) \), where \( N = \sum_{i=1}^{n} m_i \) and \( m = \max m_i \). The second one (called the breadth-K search method) obtains an approximate solution within an arbitrarily specified (worst case) error bound \( \epsilon \times 100\% \) in \( O(m \Gamma^{1/4}(\epsilon^{-2}N) \log N^{\Gamma^{1/4}(\epsilon^{-2})}) \) time.

1. Introduction

The multiple-choice continuous knapsack problem \( P \) is described as follows:

\[
P: \text{maximize } z = \sum_{i=1}^{n} \sum_{j=1}^{m_i} a_{ij} x_{ij} \]
subject to \( \sum_{i=1}^{n} \sum_{j=1}^{m_i} a_{ij} x_{ij} \leq b \), \( 0 \leq x_{ij} \leq 1 \), \( i = 1, 2, \ldots, n \) and \( j = 1, 2, \ldots, m_i \)
At most one of \( x_{i1}, x_{i2}, \ldots, x_{im_i} \) is positive for \( i = 1, 2, \ldots, n \),

where \( n, m_i (i = 1, 2, \ldots, n) \) are positive integers, and \( a_{ij} (i = 1, 2, \ldots, n, j = 1, 2, \ldots, m_i) \) are nonnegative integers. Let
Multiple-Choice Knapsack Problems

This problem was first studied in [6]. One exact algorithm based on branch-and-bound principle and two approximate algorithms (one of them, breadth-1 search method, was only outlined in [6] and its details are described in this paper) were therein proposed and tested. Their computational results are quite encouraging. For example, exact optimal solutions for problems with \( n = 500 \) and \( m_i = 2 \) (\( i = 1, 2, \ldots, n \)) are obtained in less than 0.2 seconds (on FACOM M190). Approximate solutions within 0.001\% of the optimal values are also obtained in less than one second (on FACOM 230/60) for randomly generated problems with \( n = 1000 \) and \( m_i = 2 \) (\( i = 1, 2, \ldots, n \)).

Generally speaking, however, (1.1) is still a difficult problem. It is reported in [6] that some specially structured problems require long computation time to obtain exact optimal solutions. In fact, it is proved to be NP-complete in Section 2 of this paper (for the implication of NP-completeness, see [1, 8]). Therefore, it is desirable to have approximate algorithms, which run in polynomial time and yet can theoretically guarantee the quality of the obtained solutions.

This paper considers two such approximate algorithms. The first is called breadth-1 search algorithm, and is described in Sections 4 and 5. It obtains in \( O(mn \log N) \) time an approximate solution with the ratio of its objective value and the exact optimal value greater than \( 3/4 \). This is then extended in Sections 6 and 7 to a breadth-K search algorithm, which is a polynomial time approximation scheme (in the sense of [4]) rather than an algorithm. It obtains an approximate solution with the ratio of its objective value and the exact optimal value greater than \( 1-\varepsilon \) for any given \( 0 < \varepsilon < 1/4 \), if an appropriate \( K \) is chosen. The time required for this computation is \( O(m \lceil \frac{1}{4}(e - \varepsilon^2) \rceil \), where \( \lceil x \rceil \) denotes the smallest integer not smaller than \( x \).

The discrete version of (1.1), i.e., \( P \) with (1.1c) replaced by

\[
(1.3) \quad x_{ij} = 0 \text{ or } 1, \quad i = 1, 2, \ldots, n \text{ and } j = 1, 2, \ldots, m_i
\]

has also been studied in various papers such as [3, 11, 12]. Some properties proved in these papers can also be extended to our continuous problems. [3, 11] discuss approximate algorithms for the discrete problem. Some discussion will be given in Section 8 on the use of such algorithms of the discrete version for our problem.

In [3, 6], it is noted that the following assumptions can be made without loss of generality.

\[
(1.2) \quad n = \sum_{\ell=1}^m m_\ell, \quad m = \max\{m_\ell \mid \ell = 1, 2, \ldots, n\}.
\]
\[(1.4) \quad a_{i1} \geq a_{i2} \geq \ldots \geq a_{im_i} \geq 0, \quad i = 1, 2, \ldots, n,\]
\[(1.5) \quad a_{i1} > a_{i2} > \ldots > a_{im_i} > 0, \quad i = 1, 2, \ldots, n,\]
\[(1.6) \quad \frac{a_{im_i}}{a_{im_i} - 1} > \ldots > \frac{a_{i1}}{a_{i1} - 1} > 0, \quad i = 1, 2, \ldots, n,\]
\[(1.7) \quad \sum_{i=1}^{n} a_{i1} > b.\]

Here the denominators of (1.6) are allowed to be 0 with the convention that
\[(1.8) \quad \frac{a}{0} > \frac{a'}{0} \quad \text{for any} \quad a', a', c > 0,\]
\[\frac{a}{0} > \frac{a'}{0} \quad \text{if and only if} \quad c \geq c' \quad \text{for any} \quad c, c' > 0.\]

The following property proved in [6] is also helpful to understand the structure of \(P\).

\[(1.9) \quad P \text{ has an optimal solution such that at most one variable } x_{i,j} \text{ satisfies } 0 < x_{i,j} < 1.\]

2. NP-Completeness of \(P\)

The NP-completeness of \(P\) defined in (1.1) is proved from the NP-completeness of the following ordinary 0-1 knapsack problem (e.g., [1, 8]):

\[(2.1) \quad \begin{align*}
\text{maximize} & \quad \sum_{i=1}^{n} \gamma_i x_i \\
\text{subject to} & \quad \sum_{i=1}^{n} \alpha_i x_i \leq \beta \quad \text{for } i = 1, 2, \ldots, n, \quad x_i = 0 \text{ or } 1,
\end{align*}\]

where \(\gamma_i (i = 1, 2, \ldots, n), \alpha_i (i = 1, 2, \ldots, n)\) and \(\beta\) are given positive integers. We associate the following multiple-choice continuous knapsack problem \(P_Q\) with the above \(Q\).

\[(2.2) \quad \begin{align*}
\text{maximize} & \quad \sum_{i=1}^{n} (R + \gamma_i) x_{i1} + \sum_{i=1}^{n} Rx_{i2} \\
\text{subject to} & \quad \sum_{i=1}^{n} \alpha_i x_{i1} \leq \beta, \quad 0 < x_{i,j} \leq 1, \quad i = 1, 2, \ldots, n \quad \text{and} \quad j = 1, 2, \ldots, m_i, \\
& \quad x_{i1} x_{i2} = 0, \quad i = 1, 2, \ldots, n, \quad m_i = 2, \quad i = 1, 2, \ldots, n, \quad \text{and} \quad R \text{ is a positive integer.}
\end{align*}\]
Lemma 2.1. Let \( \gamma = \max_i \gamma_i \), \( \alpha = \max_i \alpha_i \), and \( R > \gamma(a-1) \). Then \( P_Q \) has an optimal solution \( x^0 = (x^0_{i,j}) \) such that \( x^0_{i,j} = 0 \) or 1 for \( i = 1, 2, \ldots, n \) and \( j = 1, 2 \).

Proof: Let \( x^0 \) be an optimal solution of \( P_Q \) satisfying property (1.9), i.e.,
\[
0 < x^0_{i,1} \leq 1 \quad \text{and} \quad x^0_{i,j} = 0 \text{ or } 1 \quad \text{for } i \neq i_1 \text{ and } j = 1, 2.
\]
(The case of \( 0 < x^0_{i,2} \leq 1 \) can be similarly treated.) Then \( x^0_{i,1} \) satisfies
\[
\alpha_i x^0_{i,1} = b - \sum_{j \neq 1} \alpha_i x^0_{i,j} \quad (\equiv b')
\]
Since \( b' \) is an integer, \( x^0_{i,1} \) is equal to one of the followings.
\[
1 \quad \frac{2}{\alpha_i}, \quad \ldots, \quad \frac{1}{\alpha_i}
\]
Now let \( x' \) be defined by
\[
x'_{i,1} = 0, \quad x'_{i,2} = 1, \quad \text{and } x'_{i,j} = x^0_{i,j} \quad \text{for } i \neq i_1 \text{ and } j = 1, 2.
\]
\( x' \) is obviously feasible, and the optimality of \( x^0 \) implies
\[
(R + \gamma_i) x^0_{i,1} \geq R,
\]
namely
\[
x^0_{i,1} \geq \frac{R}{R + \gamma_i} \geq \frac{R}{R + \gamma} \geq \frac{\gamma(a-1)}{\gamma(a-1) + \gamma} = \alpha \geq \frac{\alpha_i - 1}{\alpha_i}
\]
This implies \( x^0_{i,1} \geq \frac{\alpha_i - 1}{\alpha_i} \) by (2.4).

Lemma 2.2. Let \( R > \gamma(a-1) \) hold in \( P_Q \). Then a solution \( x^0 = (x^0_{i,j}) \) of \( P_Q \) satisfying \( x^0_{i,j} = 0 \) or 1 \((i = 1, 2, \ldots, n \text{ and } j = 1, 2)\) is optimal if and only if \((x^0_1, x^0_2, \ldots, x^0_n)\) defined by \( x^0_{i,j} = x^0_{i,1} \quad (i = 1, 2, \ldots, n) \) is an optimal solution of \( Q \).

Proof: Note first that \( x^0_{i,1} + x^0_{i,2} = 1 \quad (i = 1, 2, \ldots, n) \) can be assumed without loss of generality since \( x^0_{i,2} \) can be set to 1 without affecting the feasibility if \( x^0_{i,1} = 0 \). Then we have
\[
\sum_{i=1}^n (R + \gamma_i) x^0_{i,1} + \sum_{i=1}^n R x^0_{i,2}
\]
\[
= nR + \sum_{i=1}^n \gamma_i x^0_{i,1}
\]
\[
= nR + \sum_{i=1}^n \gamma_i x^0_{i,1} \quad \text{(where } x^0_{i,j} = x^0_{i,1}, \quad i = 1, 2, \ldots, n).\]
Thus two objective functions in \( Q \) and \( P_Q \) differ only by a constant \( nR \). Since both \( Q \) and \( P_Q \) have the same constraint, and \( P_Q \) has an optimal solution with
Theorem 2.3. The multiple-choice continuous knapsack problem $P$, defined in (1.1), is NP-complete. (See [1, 8] for the precise definition of the NP-completeness.)

Proof: Follows from Lemmas 2.1 and 2.2, since $P \in NP$ is obvious and the 0-1 knapsack problem $Q$, which is known to be NP-complete, is transformed to a special case of $P$ (i.e., $P$ is NP-hard).

However, $P$ is not NP-complete in the strong sense (defined in [4]). This is because a pseudo polynomial algorithm for $P$ can be constructed in a manner similar to the ordinary 0-1 knapsack problem by making use of dynamic programming.

3. LP Relaxation of $P$ and its Dual

In order to develop approximate algorithms in the subsequent sections, we introduce the LP (linear Programming) relaxation $\bar{P}$ of $P$ and its dual problem. We list here only necessary results; see [6] for the detailed discussion.

$\bar{P}$: maximize $z = \sum_{i=1}^{n} \sum_{j=1}^{m} \alpha_{i,j} x_{i,j}$
subject to $\sum_{j=1}^{m} \sum_{j=1}^{m} \alpha_{i,j} x_{i,j} \leq b$

$x_{i,j} \geq 0, i = 1, 2, \ldots, n, j = 1, 2$.

The LP dual of $\bar{P}$ is as follows.

$\bar{D}$: minimize $\nu = \sum_{i=1}^{n} \mu_{i}^{\prime} + b \lambda$
subject to $\mu_{i}^{\prime} \geq \sum_{j=1}^{m} \alpha_{i,j} x_{i,j}, i = 1, 2, \ldots, n, j = 1, 2, \ldots, m_{j}$

$\mu_{i}^{\prime} \geq 0, i = 1, 2, \ldots, n$

$\lambda \geq 0$.

For a given $\lambda$, the optimal value $\bar{\nu}(\lambda)$ of $\bar{D}$ is trivially given by

$\bar{\nu}(\lambda) = \sum_{i=1}^{n} \mu_{i}^{\prime}(\lambda) + b \lambda$

$\bar{\mu}_{i}^{\prime}(\lambda) = \max\{0, \max\{c_{i,j} - a_{i,j} \lambda \mid j = 1, 2, \ldots, m_{j}\}, i = 1, 2, \ldots, n\}$.

Thus $\bar{D}$ is solved by finding a $\lambda$ that minimizes $\bar{\nu}(\lambda)$. 

Copyright © by ORSJ. Unauthorized reproduction of this article is prohibited.
From (3.3), we see that an index \( \hat{j}_k \) is relevant to determining \( \bar{u}_i(\lambda) \) only if

\[
\sigma_{\hat{j}_k} - a_{\hat{j}_k} \lambda > \sigma_{\hat{j}_k} - a_{\hat{j}_k} \lambda, \quad \hat{j} \neq j_k
\]

holds for some \( \lambda \geq 0 \). Such an index \( \hat{j}_k \) is called dominant and the set of dominant indices for each \( i \) is denoted by

\[
J_i = \{ \hat{j}_1, \hat{j}_2, \ldots, \hat{j}_{d_i} \},
\]

where \( \hat{j}_1 = 1 \) and \( \hat{j}_{d_i} = m_i \) always hold by (1.5) and (1.6). The following lemma is proved in Appendix A.

Lemma 3.1. \( J_i \) for a given \( P \) is computed in \( O(m_i) \) time for each \( i = 1, 2, \ldots, n \). Thus the total computation time for all \( i \) is \( O(N) \).

It is known [3, 6] that indices in \( J_i \) satisfy

\[
\rho_i(j_1, j_2) < \rho_i(j_2, j_3) < \ldots < \rho_i(j_{d_i - 1}, j_{d_i}) \leq \frac{\sigma_{d_i}}{\alpha_{d_i}}
\]

for \( i = 1, 2, \ldots, n \), where

\[
\rho_i(p, q) = \frac{\sigma_p - \sigma_q}{\alpha_p - \alpha_q}.
\]

For notational simplicity, however, we assume in the subsequent discussion (unless otherwise stated) that

\[
J_i = \{ 1, 2, \ldots, m_i \}, \quad i = 1, 2, \ldots, n
\]

and hence

\[
0 \leq \rho_i(1, 2) \leq \rho_i(2, 3) \leq \ldots \leq \rho_i(m_i - 1, m_i) \leq \frac{\sigma_{m_i}}{\alpha_{m_i}}.
\]

Now let

\[
L_i = \{ \rho_i(1, 2), \rho_i(2, 3), \ldots, \rho_i(m_i - 1, m_i), \sigma_{m_i} / \alpha_{m_i} \},
\]

\[
L = \bigcup_{i=1}^{n} L_i = \{ \beta_1, \beta_2, \ldots, \beta_n \}.
\]

where

\[
\beta_1 \leq \beta_2 \leq \ldots \leq \beta_n.
\]

(Some of \( \beta_i \)'s are possibly \( \infty \); the order of these \( \beta_i \)'s is determined by following the convention (1.8).) Optimal solutions of \( \bar{D} \) and \( \bar{P} \) are respectively obtained by the following algorithm [6].

The algorithm starts with \( \lambda = \infty \) (for \( \bar{D} \)) and \( x = (00 \ldots 0) \) (for \( \bar{P} \)), and sweeps list \( L \) from right to left in the order \( \beta_n, \beta_{n-1}, \ldots, \beta_1 \). Upon crossing
\[ \beta_k \text{ in this process, } \lambda \text{ is set to } \beta_k \text{ and } x \text{ is modified as follows.} \]

(a) \( x_{im} \) is changed from 0 to 1 if \( \beta_k = \frac{a_{im}}{a_{im}} \)

(b) \( x_{ij} \) is set to 0 but \( x_{ij-1} \) is set to 1 if \( \beta_k = \rho_i(j-1, j) \).

Index \( j \) memorizes \( j \) satisfying \( x_{ij} = 1 \) for each \( i \).

If the new solution \( x \) does not satisfy constraint (1.1B) for \( \beta_k = \beta_k \) (this is checked by \( v' \)), the computation terminates.

Algorithm DUAL(P)

\[ \text{D1. Obtain } L \text{ as shown in (3.11) and (3.12).} \]

\[ \text{D2. } v' + b, j'_i \rightarrow \infty \text{ for } i = 1, 2, \ldots, n, k + N. \]

\[ \begin{align*}
\text{D3.} & \quad v' \left\{ \\
& \quad v' - a_{im} \quad \text{if } \beta_k = \frac{a_{im}}{a_{im}} \\
& \quad v' - (a_{ij-1} - a_{ij}) \quad \text{if } \beta_k = \rho_i(j-1, j),
\end{align*} \]

\[ \begin{align*}
\text{D4.} & \quad j'_{i} \rightarrow \left\{ \\
& \quad m_i \quad \text{if } \beta_k = \frac{a_{im}}{a_{im}} \\
& \quad j-1 \quad \text{if } \beta_k = \rho_i(j-1, j).
\end{align*} \]

\[ \text{D5. } k + k, \lambda_{i} = \beta_k, \quad \text{and halt.} \]

The computed \( k = \bar{k} \) defines \( \lambda = \beta_k \) that minimizes \( \bar{V}(\lambda) \) of (3.3), i.e., \( \bar{V}(\lambda) \) is the optimal value of \( \bar{D} \). Denote \( j'_i \) computed by DUAL(P) by \( j'_i(\bar{k}) \). Let \( p(i) \) denote the smallest \( p \geq \bar{k} \) for each \( i \), with \( \beta_p = \rho_i(j-1, j) \) or \( a_{im}/a_{im} \). Then \( j'_i(\bar{k}) \) and \( \bar{e} \) satisfy

\[ j'_i(\bar{k}) = \\
\begin{align*}
& m_i \quad \text{if } \beta_p(i) = \frac{a_{im}}{a_{im}} \\
& j-1 \quad \text{if } \beta_p(i) = \rho_i(j-1, j) \\
& \infty \quad \text{if } p(i) \text{ does not exist.}
\end{align*} \]

\[ \bar{e} = b - \bar{\sum}_{i \in E} a_{ij} \bar{k}, \quad \text{where } \bar{e} \text{ satisfies} \]

\[ \beta_k = \frac{a_{im}}{a_{im}} \text{ or } \beta_k = \rho_i(j-1, j). \]

An optimal solution \( \bar{x} = (\bar{x}_{11}, \bar{x}_{12}, \ldots, \bar{x}_{nm}) \) of \( \bar{P} \) is given by

\[ \begin{align*}
\text{D4.} & \quad \text{If } v' \leq 0, \text{ go to D5; else return to D3 after letting } k + k - 1. \\
\text{D5.} & \quad k + k, \lambda + \beta_k, \quad \text{and halt.} \\
\text{D4.} & \quad \text{If } v' \leq 0, \text{ go to D5; else return to D3 after letting } k + k - 1.
\end{align*} \]

\[ \begin{align*}
\text{D5.} & \quad k + k, \lambda + \beta_k, \quad \text{and halt.} \\
\text{D4.} & \quad \text{If } v' \leq 0, \text{ go to D5; else return to D3 after letting } k + k - 1.
\end{align*} \]

\[ \begin{align*}
\text{D5.} & \quad k + k, \lambda + \beta_k, \quad \text{and halt.} \\
\text{D4.} & \quad \text{If } v' \leq 0, \text{ go to D5; else return to D3 after letting } k + k - 1.
\end{align*} \]

\[ \begin{align*}
\text{D5.} & \quad k + k, \lambda + \beta_k, \quad \text{and halt.} \\
\text{D4.} & \quad \text{If } v' \leq 0, \text{ go to D5; else return to D3 after letting } k + k - 1.
\end{align*} \]

\[ \begin{align*}
\text{D5.} & \quad k + k, \lambda + \beta_k, \quad \text{and halt.} \\
\text{D4.} & \quad \text{If } v' \leq 0, \text{ go to D5; else return to D3 after letting } k + k - 1.
\end{align*} \]

\[ \begin{align*}
\text{D5.} & \quad k + k, \lambda + \beta_k, \quad \text{and halt.} \\
\text{D4.} & \quad \text{If } v' \leq 0, \text{ go to D5; else return to D3 after letting } k + k - 1.
\end{align*} \]

\[ \begin{align*}
\text{D5.} & \quad k + k, \lambda + \beta_k, \quad \text{and halt.} \\
\text{D4.} & \quad \text{If } v' \leq 0, \text{ go to D5; else return to D3 after letting } k + k - 1.
\end{align*} \]
(i) Case of $\beta_{\bar{k}} = \sigma_{\bar{l} \bar{k}} / a_{\bar{l} \bar{k}}$:

\[ \bar{x}_{i, j, \bar{k}} = 1 \text{ for } i \neq \bar{i} \text{ such that } j_{\bar{k}} < \bar{\omega} \]

\[ \bar{x}_{\bar{l} \bar{k}} = \bar{\omega} / a_{\bar{l} \bar{k}} \]

\[ \bar{x}_{i, j} = 0 \text{ for all other } i \text{ and } j \]

\[ \bar{z} = \sum_{\bar{l} \bar{k}} \sum_{i, j} \bar{z}_{i, j, \bar{k}} = \sum_{\bar{l} \bar{k}} \sum_{i, j} c_{i, j} \bar{x}_{i, j, \bar{k}} + \sigma_{\bar{l} \bar{k}} \bar{\omega} / a_{\bar{l} \bar{k}} \]

(ii) Case of $\beta_{\bar{k}} = \rho_{\bar{l}} (\bar{i} - 1, \bar{j})$:

\[ \bar{x}_{i, j, \bar{k}} = 1 \text{ for } i \neq \bar{i} \text{ such that } j_{\bar{k}} < \bar{\omega} \]

\[ \bar{z}_{\bar{l} \bar{j}} = \theta, \quad \bar{z}_{\bar{l} \bar{j} - 1} = 1 - \theta \]

\[ \bar{x}_{i, j} = 0 \text{ for other } i \text{ and } j \]

\[ \bar{z} = \sum_{\bar{l} \bar{k}} \sum_{i, j} \bar{z}_{i, j, \bar{k}} = \sum_{\bar{l} \bar{k}} \sum_{i, j} c_{i, j} \bar{x}_{i, j, \bar{k}} + \theta \sigma_{\bar{l} \bar{j}} + (1 - \theta) \sigma_{\bar{l} \bar{j} - 1} \]

where

\[ \theta = (a_{\bar{l} \bar{j} - 1} - \bar{\omega}) / (a_{\bar{l} \bar{j} - 1} - a_{\bar{l} \bar{j}}) \]

Obviously, computation of sets of dominant indices $J_{\bar{k}}$ (see (3.5)), execution of DUAL(P) and computation of (3.15) or (3.16) together requires $O(N\log N)$ time (computation of $J_{\bar{l}}(i = 1, 2, \ldots, n)$ requires $O(N)$ (Lemma 3.1), D1 of DUAL(P) requires $O(N\log N)$ for sorting $\beta_{\bar{l}}$, $\beta_{\bar{2}}, \ldots, \beta_{\bar{n}}$, and the rest of DUAL(P) plus (3.15) or (3.16) requires $O(N)$). If $\bar{x}$ is obtained by (3.15), it satisfies condition (1.1D) and hence is also an optimal solution of $P$. On the other hand, if $\bar{x}$ is obtained by (3.16), (1.1D) is not satisfied for exactly one index $i = \bar{\omega}$ unless $\theta = 0$ holds. In case of $\theta > 0$, two variables $x_{\bar{l} \bar{j} - 1}$ and $x_{\bar{l} \bar{j}}$ assume positive values.

Remark. The $O(N\log N)$ computation time given above can be reduced to $O(N)$ by employing the fast median finding algorithm in such a manner as used in [2, 11]. The details are not given here, however, since the time requirement of this portion is dominated by others in the approximate algorithms described in the subsequent sections.

4. Approximate Solutions by Rounding

Based on the LP optimal solution $\bar{x}$ obtained by (3.16) (if (3.15) is the case, $P$ is solved), we here construct three approximate solutions of $P$, $x^{(1)}$, $x^{(2)}$ and $x^{(3)}$. These will be used in the approximate algorithms discussed in Section 5-7.
$x^{(1)}$ and $x^{(2)}$ are given as follows.

\[(4.1)\]
\[
\begin{align*}
  x^{(1)}_{i,j} &= \bar{x}_{i,j} \quad \text{for } i \neq \bar{7} \text{ and } j = 1, 2, \ldots, m_i, \\
  x^{(1)}_{j,j} &= 1, \quad x^{(1)}_{j,j} = 0 \quad \text{for } j \neq j.
\end{align*}
\]

\[(4.2)\]
\[
\begin{align*}
  x^{(2)}_{i,j} &= \bar{x}_{i,j} \quad \text{for } i \neq \bar{7} \text{ and } j = 1, 2, \ldots, m_i, \\
  x^{(2)}_{j,j-1} &= e/\alpha_{j,j-1}, \quad x^{(2)}_{j,j-1} = 0 \quad \text{for } j \neq j-1.
\end{align*}
\]

The objective values of these solutions are respectively given by

\[(4.3)\]
\[
\begin{align*}
  z^{(1)} &= \sum_{i,j} c_{i,j} x^{(1)}_{i,j} = \sum_{i \neq \bar{7}} \alpha_{i,j} \bar{x}_{i,j} + \sigma_{j,j}, \\
  z^{(2)} &= \sum_{i,j} c_{i,j} x^{(2)}_{i,j} = \sum_{i \neq \bar{7}} \alpha_{i,j} \bar{x}_{i,j} + \bar{e}((\sigma_{j,j-1}/\alpha_{j,j-1}).
\end{align*}
\]

To define the third approximate solution $x^{(3)}$, note that $\bar{7}$ and $\bar{7}-1$ given by $\beta_{\bar{7}} = \lambda = \sigma_{j,j-1}, j$ satisfy

\[(4.4)\]
\[
\alpha_{j,j-1}/\alpha_{j,j-1} > \beta_{\bar{7}}(\bar{7}-1, j)
\]
as easily proved from assumptions (1.4)-(1.6). Let $k'$ be the smallest index satisfying $\beta_{k'} \epsilon L$ (see (3.11)) and $\beta_{k'} \geq \sigma_{j,j-1}/\alpha_{j,j-1}$ (recall that $\sigma_{j,j-1}/\alpha_{j,j-1}$ is not in $L$), where $k' > \bar{7}$ by (4.4), and let

\[(4.5)\]
\[
A' = \sum_{i \neq \bar{7}} \alpha_{i,j} \bar{x}_{i,j} (k') < b,
\]

where $\bar{x}_{i,j} (k')$ is given by (3.13) with $\bar{7}$ replaced by $k'$ in its definition. Then $x^{(3)}$ is defined as follows.

1. Case of $A' + \alpha_{j,j-1} \geq b$: Then

\[(4.6)\]
\[
\begin{align*}
  x^{(3)}_{i,j}(k') &= 1 \quad \text{for } i \neq \bar{7}, \quad x^{(3)}_{i,j} = 0 \quad \text{for } i \neq \bar{7} \text{ and } j \neq j (k'), \\
  x^{(3)}_{j,j-1} &= (b - A')/\alpha_{j,j-1}, \quad x^{(3)}_{j,j-1} = 0 \quad \text{for } j \neq j-1, \\
  z^{(3)} &= \sum_{i \neq \bar{7}} \alpha_{i,j} \bar{x}_{i,j} (k') + \sigma_{j,j-1}(b - A')/\alpha_{j,j-1}.
\end{align*}
\]

2. Case of $A' + \alpha_{j,j-1} < b$: Then $x^{(3)}$ is not computed and

\[(4.7)\]
\[
z^{(3)} = -\infty.
\]

Finally, let $x^{(T)}$ denote the best solution among $x^{(1)}$, $x^{(2)}$ and $x^{(3)}$ if (3.16) is the case, and $\bar{x}$ if (3.15) is the case. Let

\[(4.8)\]
\[
\begin{align*}
  z^{(T)} &= \max[z^{(1)}, z^{(2)}, z^{(3)}] \quad \text{if (3.16) is the case for } P, \\
  z^{(T)} &= \bar{z} \quad \text{if (3.15) is the case for } P.
\end{align*}
\]

$x^{(T)}$ is called the approximate solution of $P$ obtained by rounding.

Lemma 4.1. If $A' + \alpha_{j,j-1} \geq b$ holds for $A'$ defined in (4.5), $x^{(3)}$ obtained
by (4.6) is an optimal solution of $P' \equiv P(x_{ij} = 0 \text{ for } j \neq 3-1)$ (i.e., $P$ with additional restrictions $x_{ij} = 0 \text{ for } j \neq 3-1$).

Proof: By (4.5) and (4.6), it is easily shown that

$$
\sum_{i=1}^{m} a_{ij} x_{ij}^{(3)} = b
$$

holds and hence $x^{(3)}$ is a feasible solution of $P'$ since it satisfies (1.1C) and (1.1D) as well. Next apply DUAL($P'$) to obtain an optimal solution of $P'$. $L$ for $P'$ is obtained from $L$ for $P$, by deleting $L^*_j$ (see (3.10)) and inserting $a_{ij}/a_{i3-1}$.

By condition $A' + a_{ij} \geq b$, DUAL($P'$) terminates at $\theta = a_{ij}/a_{i3-1}$ with $\tilde{\theta} = b - A' (> 0)$. Therefore $\tilde{x}$ computed for $P'$ by (3.15) is equal to $x^{(3)}$, i.e., $x^{(3)}$ is an LP optimal solution. This shows that $x^{(3)}$ is an optimal solution of $P'$.

Lemma 4.2. If $A' + a_{ij} \geq b$ holds for $A'$ of (4.5), then $z(T)$ obtained by (4.8) for $P$ satisfies

$$
z(T) / \tilde{\theta} > 3/4,
$$

where $\tilde{\theta}$ is the LP optimal value of $P$.

Proof: Assume that (3.16) holds since otherwise $z(T) / \tilde{\theta} = 1$. Let $\bar{\lambda}$ and $\bar{\lambda} = c_{i3-1}(j-1, j)$ be obtained by DUAL($P$), and let

$$
C = \sum_{i \neq j} c_{ij}(K)
$$

(i.e., the first portion in $\tilde{\theta}$ of (3.16)). We first show that

$$
z \geq (a_{ij} - \bar{\lambda})p_{i3-1}(j-1, j),
$$

where $\bar{\theta}$ is given by (3.14). (Note that $a_{ij} < \bar{\lambda} \leq a_{ij}$ holds by the termination condition of DUAL($P$).) Now note that the LP optimal solution $\bar{x}$ obtained by (3.16) satisfies (1.1D) (and hence $\bar{x}$ is an optimal solution of $P$) if $\theta = 0$ holds. In other words, $\bar{x} = (\bar{x}_{ij})$ defined by

$$
\begin{align*}
\bar{x}_{ij} &= \bar{x}_{ij} \text{ for } i \neq j, \\
\bar{x}_{ij} &= 1, \bar{x}_{ij} = 0 \text{ for } j = 1, 2, \ldots, m_i \\
\bar{x}_{ij} &= 0 \text{ for } j \neq 3-1
\end{align*}
$$

is an optimal solution of $P$ which is $P$ with the right hand side $b$ replaced by $\tilde{\theta} = b + (a_{ij} - \bar{\theta})$ (i.e., $b$ is increased by $(a_{ij} - \bar{\theta})$ to make $\theta = 0$ in (3.17)). Thus

$$
\tilde{\theta} = C + a_{ij} - \bar{\theta}
$$

is the optimal value of $\tilde{P}$. Since $\tilde{x}$ is also an LP optimal solution of $\tilde{P}$, we have

$$
\tilde{x} \leq \lambda(a_{ij} - \bar{\theta}).
$$

This follows from the LP duality theory because $\tilde{P}$ is obtained from the LP re-
laxation of $\tilde{P}$ by decreasing the right hand side by $(a_{i,j-1} - \tilde{e})$ and $\lambda$ is the value of the dual variable $\lambda$ corresponding to the constraint (1.18) for both $\tilde{P}$ and $\tilde{P}$.

The condition $A' + a_{i,j-1} \tilde{P} < b$ guarantees the property $x_{i,j-1} = 1$ in the LP optimal solution of $P' = P(x_{i,j} = 0, j \neq j-1)$ (execute DUAL($P'$) as in the proof of Lemma 4.1). Thus the LP optimal value $\tilde{z}'$ of $P'$ satisfies

(4.15)  
$\tilde{z}' \geq a_{i,j-1} - e_{i,j-1}$

Since $\tilde{z}' \leq \tilde{z}$ holds, combining (4.13), (4.14) and (4.15) yields (4.11).

Now note that $z(2) = \max[z(1), z(2)]$ holds by condition $A' + a_{i,j-1} \tilde{P} < b$. Consider $z(1)$, $z(2)$ and $\tilde{z}$ as functions of $\tilde{e}$ (see (4.3) and (3.16), where $\theta$ is determined by $\tilde{e}$ as shown in (3.17)), and denote them by $z(1)(\tilde{e})$, $z(2)(\tilde{e})$ and $\tilde{z}(\tilde{e})$. By direct calculation, it can be shown that

(a) $z(1)(\tilde{e})/\tilde{z}(\tilde{e})$ is nonincreasing in $\tilde{e}$,
(b) $z(2)(\tilde{e})/\tilde{z}(\tilde{e})$ is nondecreasing in $\tilde{e}$, and
(c) $(z(1)(\tilde{e})/\tilde{z}(\tilde{e})) = (z(2)(\tilde{e})/\tilde{z}(\tilde{e}))$ if and only if

(4.16)  
$\tilde{e} = a_{i,j-1} - e_{i,j-1}$

holds. This proves that $z(2)/\tilde{z} = \max[z(1)/\tilde{z}, z(2)/\tilde{z}]$ takes its minimum when

(4.17)  
$\tilde{e} = a_{i,j-1} - e_{i,j-1}$

Then

(4.18)  
$\frac{z(2)}{\tilde{z}} \geq \frac{z(1)(a_{i,j-1} - e_{i,j-1})}{\tilde{z}(a_{i,j-1} - e_{i,j-1})} = \frac{C_0 + \theta C_x}{C_0 + (1-\theta)C_x}$ (where $\theta$ is given by (4.17))

where

(4.19)  
$\alpha = \frac{C_x}{a_{i,j-1}}$, $\beta = \frac{C_x}{a_{i,j-1}}$, and $1 \geq \alpha > \beta > 0$.

The last formula of (4.18) takes its minimum $(\beta + 3)/4$ when $\alpha = (\beta + 1)/2$ holds. $(\beta + 3)/4$ is greater than $3/4$ because $0 < \beta < 1$. □
In closing this section, we show that $x^{(T)}$ is computed in $O(N)$ time after DUAL($P$) is executed. $x^{(1)}$ and $x^{(2)}$ are trivially obtained in $O(N)$ time. To obtain $A'$ of (4.5), DUAL($P$) is slightly modified. First

$$(4.20) \quad c_{s,j-1}/a_{s,j-1}, \quad i=1, 2, \ldots, n \text{ and } j=2, 3, \ldots, m_i$$

are added to list $L = \{\beta_1, \beta_2, \ldots, \beta_N\}$ and sorted together with $\beta_1, \beta_2, \ldots, \beta_N$. Whenever $\beta_{k}<c_{s,j-1}/a_{s,j-1}$ is detected in step D3 of DUAL($P$),

$$(4.21) \quad A' = b - v' - a_{s,j-1} = \sum_{l \neq s} a_{s,j-1}$$

is computed and stored with a pointer to $c_{s,j-1}/a_{s,j-1}$. This modification does not change the order of time required by DUAL($P$). The computation of $x^{(3)}$ by (4.6) is then done in $O(N)$ time.

5. Approximate Algorithm by Breadth-1 Search

The approximate algorithm by breadth-1 search first obtains the LP optimal solution $\bar{x}$ of $P$ by applying DUAL($P$). If $\bar{x}$ is feasible in $P$ (i.e., (3.15) holds or $\theta=0$ in (3.16)), then $P$ is solved. Otherwise (i.e., (3.16) with $\theta>0$), $x^{(T)}$ is calculated and a new problem $P(x^{(T)}; x_{h+1}^-, \beta_{h+1}=0)$ is generated, where $x_{h+1}^-$ is determined by $x_{h+1}^-$ obtained in DUAL($Q_h$). Return to A2 after letting $h+h+1$.

The LP optimal value $\bar{z}_1$ obtained for $Q_1$ gives an upper bound on the exact optimal value $z^0$ of $P$, i.e., $z^{(B)} \leq z^0 \leq \bar{z}_1$. 

Copyright © by ORSJ. Unauthorized reproduction of this article is prohibited.
Theorem 5.1. \( z(B) \) obtained by APPRXB1\((P) \) satisfies
\[
\frac{z(B)}{z^0} > \frac{3}{4},
\]
where \( z^0 \) is the exact optimal value of \( P \).

Proof: Assume that APPRXB1\((P) \) has solved \( Q_h', \ h=1, 2, \ldots, H, \) i.e., the LP optimal solution of \( Q_H \) is feasible in \( Q_H' \). Note that for each \( h \) satisfying \( 1 \leq h \leq H-1 \), the optimal value of \( Q_h \) is equal either to the optimal value of \( Q_h'(x_{hj}=0, j \neq j_h-1) \) (denoted by \( Q^*_{h+1} \)). Now assume that condition \( A' + a_{h-1} \geq b \) of (4.6) holds for \( Q_1, Q_2, \ldots, Q_g \) (\( 0 \leq g \leq H-1 \) because \( Q_H \) is solved by DUAL\((Q_H) \)). Then optimal solutions of \( Q_1', Q_2', \ldots, Q_{g+1}' \) are respectively obtained as \( z(3) \) by Lemma 4.1. Therefore if one of \( Q_1', Q_2', \ldots, Q_{g+1}' \) contains an optimal solution of \( P \), \( P \) is solved by APPRXB1\((P) \), i.e., \( z(B)/z^0 = 1 \). On the other hand, if none of \( Q_1', Q_2', \ldots, Q_{g+1}' \) contains an optimal solution of \( P \), \( Q_{g+1}' \) contains an optimal solution of \( P \).

Denoting the optimal value of \( Q_{g+1}' \) by \( z_{g+1}' \), we have
\[
\frac{z(B)}{z^0} \geq \frac{z(T)}{z^0} \geq \frac{z(T)}{z_{g+1}^0} \geq \frac{z(T)}{z_{g+1}^0} > \frac{3}{4},
\]
where the last inequality follows from Lemma 4.2. \( \Box \)

By carefully analyzing the above proof, we notice that the execution of APPRXB1 may be cut off at \( B_2 \) as soon as \( A' + a_{h-1} < b \) holds for the current \( Q_h \). This does not change the worst case bound of Theorem 5.1. Our APPRXB1, however, seems to give better performance in the sense of the average quality of solutions, without increasing the order of computation time.

Theorem 5.2. Let \( z(B) \) and \( z^0 \) be defined as in Theorem 5.1. The bound obtained in Theorem 5.1 is sharp in the sense that there exists a problem \( P \) (defined by (1.1)) such that
\[
\frac{z(B)}{z^0} < \left( \frac{3}{4} \right) + \delta
\]
holds for any given \( \delta > 0 \).

Proof: Consider \( P \) with \( n=2, m_1=m_2=2 \) and the following coefficients:
\[
\begin{align*}
\alpha_{11} &= \alpha + 1, \quad \alpha_{12} = 1, \quad \alpha_{21} = \frac{(\alpha + 2)}{2\beta} + \frac{(\alpha + 1)}{2}, \quad \alpha_{22} = \frac{2(\alpha + 2)}{2\beta} + \frac{(\alpha + 1)}{2} \\
\beta &= \frac{(\alpha + 1)}{2} \quad + \frac{a_{01}}{2}, \quad \beta_{21} = \frac{(\alpha + 2)}{2\beta} + \frac{(\alpha + 1)}{2} \\
\alpha_{11} &= \frac{2\beta}{2(\alpha + 2)}, \quad \alpha_{12} = \beta, \quad \alpha_{21} = \frac{\beta}{2\alpha} + \frac{(\alpha + 1)}{2}, \quad \alpha_{22} = 1,
\end{align*}
\]
where \( \alpha \) and \( \beta \) are parameters satisfying \( \alpha > \beta > 1 \). First note that list \( L \) of (3.11) is given by
\[
L = \{ \beta_1, \beta_2, \beta_3, \beta_4 \} = \{ \rho_1(1, 2), \rho_2(1, 2), \alpha_{22}/\alpha_{22}, \alpha_{12}/\alpha_{12} \},
\]

Copyright © by ORSJ. Unauthorized reproduction of this article is prohibited.
where
\[ \rho_1(1, 2) = \rho_2(1, 2) = \beta/a < c_{22}/a_{22} = 2\beta/(a+2) < c_{12}/a_{12} = \beta. \]

Applying DUAL(Q_1), where Q_1 = P, we obtain
\[
\begin{align*}
\bar{\lambda} &= 1, \quad \bar{\xi} = \beta, \quad \rho_2(\bar{\xi}, \bar{\lambda}) = \rho_1(1, 2) = \beta/a, \\
\bar{e} &= b - a_{21} = (a+1)/2.
\end{align*}
\]
\[ (5.4) \]
\[
\begin{pmatrix}
\bar{x}_1 \\
\bar{x}_2
\end{pmatrix}
= 
\begin{pmatrix}
(x_{11}, x_{12}, x_{21}, x_{22})
\end{pmatrix}
= 
\begin{pmatrix}
(38/2) + (8/2a) + 1, \\
(38/2) + (8/2a) + 1, \\
(38/2) + (8/2a) + 1, \\
(38/2) + (8/2a) + 1
\end{pmatrix}
= 
\begin{pmatrix}
(\beta) + (a+1)/2a, \\
(\beta) + (a+1)/2a, \\
1, \\
0
\end{pmatrix}
= 
\bar{x}_1.
\]

Since \( \bar{x}_1 \) is not feasible, \( x(1), x(2) \) and \( x(3) \) are computed for \( Q_1 \):
\[
\begin{align*}
x(1) &= (0, 1, 1, 0), \quad z(1) = (38/2) + (8/2a) + 1, \\
x(2) &= (1/2, 0, 1, 0), \quad z(2) = (38/2) + (8/2a) + 1, \\
x(3) &= \text{not computed, and } z(3) = -\infty,
\end{align*}
\]
\[
\begin{pmatrix}
x_1 \\
x_2 \\
x_3
\end{pmatrix}
= 
\begin{pmatrix}
(38/2) + (8/2a) + 1
\end{pmatrix}
= 
\bar{x}_1.
\]

The result for \( x(3) \) follows since \( k' = 4 \) (because \( c_{11}/a_{11} = 2\beta/(a+1) \) is located between \( \beta_3 = 2\beta/(a+2) \) and \( \beta_4 = \beta \) in \( L \)), \( A' = 0 \) and \( A' + \gamma_{a-1} = a_{11} = a+1 < b = ((a+2)/2\beta) + (a+1) \) hold in \((4.5)-(4.7)\).

According to APPRXBI(P), \( Q_2 = P(x_{11} = 0) = P(x_{11} = 0) \) is next examined. However, \( Q_2 \) does not satisfy \((1.7)\) because
\[
\begin{align*}
a_{12} + a_{21} &= ((a+2)/2\beta) + ((a+1)/2) + 1, \\
< b &= ((a+2)/2\beta) + (a+1),
\end{align*}
\]
and hence \( x_{12} = x_{21} = 1, x_{22} = 0 \) is an optimal solution as noted in the footnote given to D4 of DUAL(P). The optimal value is \( z_2^0 = a_{12} + a_{21} = (\beta(a+1)/2a) + 1 + \beta = z(1) \). This is stored as \( z_2^1 \). APPRXBI(P) then terminates. \( z(B) \) is given by
\[
\begin{align*}
z(B) &= \max[z_1^T, z_2^T] \\
&= (38/2) + (8/2a) + 1.
\end{align*}
\]

On the other hand, it is easy to show that
\[
\begin{align*}
x^0 &= (1, 0, 0, 1), \quad z^0 = 2\beta + 1(= \bar{x}_1), \\
z(B)/z^0 &= (38/2) + (8/2a) + 1/(2\beta + 1) > 3/4.
\end{align*}
\]

The left hand side approaches to \( 3/4 \) when both \( \alpha \) and \( \beta \) tend to infinity in such a way that \( \beta/\alpha \to 0 \) holds.

Although the problem \( P \) in the above proof has \( n = 2 \) and \( m_1 = m_2 = 2 \), it can
be easily generalized to a problem with arbitrary \( n \) and \( m_i \geq 2 \) \((i = 1, 2, \ldots, n)\) by introducing coefficients such that all the new \( \rho_i(J-1, j) \) and \( c_i l m_i / a l m_i \) are sufficiently smaller than \( \rho_1(1, 2) \) of (5.3). Then these quantities are located to the left of \( \rho_1(1, 2) \) in list \( L \), and the computation of \( \text{APPRXXB1} (P) \) is not substantially affected by this modification.

Now we estimate the computation time required to carry out \( \text{APPRXXB1} (P) \).

For each \( Q_{h+1} = Q_h (x_{h+1} = 0) \) encountered in \( B_2 \), sets of dominant indices \( J_i \) \((i = 1, 2, \ldots, n)\) (see (3.5)) are first computed. Since \( J_i \)'s for \( i \neq j \) do not change from the previous \( J_i \) for \( Q_h \), only \( J_i \) is actually updated in \( O(m_j \) time (Lemma 3.1). Let \( L_i^h \) and \( L_i \) denote \( L_i^h \) and \( L \) for \( Q_h \) respectively (see (3.10), (3.11)). As a result of changing \( J_i \), \( L_i^h \) in \( L \) is replaced with \( L_i^h+1 \) to obtain \( L_i^h+1 \). To perform this process efficiently, a master list \( \tilde{L} \) is formed before solving \( Q_1 = P \) in \( B_2 \). \( \tilde{L} \) consists of all the elements of the form

\[
\rho_i (J_i, j), \quad i = 1, 2, \ldots, n \text{ and } 1 \leq j_1 \leq j_2 \leq m_i
\]

(5.7)

and these elements are sorted in the nondecreasing order. Since \( \tilde{L} \) contains

\[
\sum_{l=1}^{m} m_i' (m_i' - 1) / 2 + \sum_{l=1}^{m} m_i' \leq mN + N
\]

elements, its requires \( O(mN \log N) \) time (note that \( mN \leq N^2 \)). However \( L_1 \) (list \( L \) for \( Q_1 \)) can be formed from \( \tilde{L} \) in \( O(mN) \) time if appropriate links are added to \( \tilde{L} \). Furthermore, each update from \( L_i^h \) to \( L_i^h+1 \) is done in \( O(m_i) \) time because at most \( O(m_i) \) elements are involved each time in replacing \( L_i^h \) with \( L_i^h+1 \). Thus the total time required for computing \( L_i^h \) for \( h = 1, 2, \ldots, H \) in \( \text{APPRXXB1} (P) \) is \( O(mN) \) since \( H \leq N \) holds as noted prior to \( \text{APPRXXB1} (P) \), plus the extra \( O(mN \log N) \) time required for sorting \( \tilde{L} \).

In order to obtain \( \tilde{x}_h \) or \( x_h^{(P)} \) for each \( Q_h \), \( \text{DUAL}(Q_h) \) is executed for \( h = 1, 2, \ldots, H \). \( \text{DUAL}(Q_1) \) searches \( \tilde{x} \) on \( L_1 \) starting from the right end of \( L_1 \) to its left. To solve \( Q_h \) for \( h \geq 2 \), however, it is not necessary to repeat the search over again, by the next lemma.

Lemma 5.3. Let \( \tilde{x}_h \) denote \( \tilde{x} \) obtained as a result of executing \( \text{DUAL}(Q_h) \), and let \( L_i^h \) and \( L_i^h+1 \) be defined as above. Then any element \( \beta \) which is in exactly one of \( L_i^h \) and \( L_i^h+1 \) \((i.e., \text{those which are removed or inserted when } L_i^h+1 \text{ is obtained from } L_i^h)\) satisfies

\[
\beta \leq \tilde{x}_h
\]

(Proof is given in Appendix B.)

Namely, \( L_2 \) is formed from \( L_1 \) without changing the elements located to the
right of $\lambda_1 = \beta_{11}$. Thus the search for the new $\lambda_2$ can be resumed on $L_2$ from the element, which was next to the right of $\beta_{11}$ in $L_1$, to its left. This can be repeated for $h = 2, 3, ..., H$. When APPRXBL$(P)$ halts (i.e., $Q_H$ is solved), we have list $L_H$ and its length is $O(N)$. Thus the total time required for search $\lambda_1$, $\lambda_2$, ..., $\lambda_H$ is $O(N)$. Although we omit the details, computation of $\overline{x}_h$ and $x_h^{(T)}$ for all $h = 1, 2, ..., H$ can also be done in $O(N)$ time since we need only to compute how $\overline{x}_h$ and $x_h^{(T)}$ differ from $\overline{x}_{h-1}$ and $x_{h-1}^{(T)}$.

Consequently we have the next theorem.

Theorem 5.4. APPRXBL$(P)$ can be implemented so that the required time is $O(mN\log N)$ for sorting $\overline{x}$ and $O(mN)$ for the rest of computation.

As noted in Introduction, the average computational results reported in [6] are extremely good, and breadth-1 search seems to be sufficient for practical purposes. Breadth-$K$ search discussed in the following sections may be mainly of theoretical importance.

6. Approximate Algorithm by Breadth-$K$ Search

During computation of breadth-$K$ search algorithm, which will be described below, partial problems $P_k = P(D_k, E_k, F_k)$ are generated, where $D_k$, $E_k$ and $F_k$ are some subsets of the set of all index pairs $I = \{(i, j) \mid i = 1, 2, ..., n, j = 1, 2, ..., m_i\}$. $D_k$, $E_k$ and $F_k$ have the following interpretation:

\[(i, j) \in D_k \text{ implies that } x_i^j = 1 \text{ holds in } P_k:\]
\[(i, j) \in E_k \text{ implies that } x_i^j = 0 \text{ holds in } P_k:\]
\[(i, j) \in F_k \text{ implies that } 0 \leq x_i^j \leq 1 \text{ holds in } P_k \text{ and } x_i^j = 0 \text{ hold for all } j' \neq j.\]

Furthermore,

\[(6.2) \quad P_k = \emptyset \text{ or } \{(i, j)\} \text{ (a singleton)}\]

is assumed by property (1.9). Problem $P_k = P(D_k, E_k, F_k)$ is actually constructed from $P$ as follows:

(i) Delete all pairs $(i, j')$($j' = 1, 2, ..., m_i$) from $I$ if $(i, j) \in D_k$ for some $j$, delete $(i, j)$ if $(i, j) \in E_k$, and delete $(i, j')$($j' \neq j$) if $(i, j) \in F_k$.

Let the resulting set of index pairs be $G_k$.

(ii) $P_k = P(D_k, E_k, F_k)$ is defined by

\[
\text{maximize } z_k = \sum_{(i, j) \in D_k} a_{ij}^k \sum_{(i, j) \in E_k} \sum_{(i, j) \in F_k} g_{ij}^k x_{ij}^k.
\]

Copyright © by ORSJ. Unauthorized reproduction of this article is prohibited.
subject to \( \sum_{(i,j) \in G_k} a_{i,j} x_{i,j} \leq b_k \)
\[ 0 \leq x_{i,j} \leq 1, \text{ for } (i, j) \in G_k, \]
for at most one of \( x_{i,j}((i, j) \in G_k) \) is positive for each \( i, \)

where
\[
 b_k = b - \sum_{(i,j) \in D_k} a_{i,j}.
\]

Breadth-\( K \) search algorithm, where \( K \) is a given positive integer, is a variant of branch-and-bound algorithms. During computation, \( \mathcal{A} \) stores all the active partial problems, i.e., \( P_k \)'s which have been generated but not tested yet. The computation halts when \( \mathcal{A} \) becomes empty. The best feasible solution of \( P \) obtained by then is the computed approximate solution \( x^{(A)} \) of \( P \) with its value \( z^{(A)} \). \( z \) is used to maintain the best objective value during computation.

Algorithm APPRXBK\((P, K)\)

Remark. \( K \) is a positive integer.

A1. \( \mathcal{A} = \{P\}, z = 0 \) and \( z^+ = \infty \).

A2. If \( \mathcal{A} = \phi \), let \( x^{(A)} = x, z^{(A)} = z \) and halt. If \( \mathcal{A} \neq \phi \), select a partial problem \( P_k = P(D_k, E_k, F_k) \in \mathcal{A} \).

A3. Obtain \( z_k^-(\text{LP optimal value}) \), the approximate solution \( x_k^+(B) \) and its value \( z_k^-(B) \) of \( P_k \) by executing APPRXB1\((P_k)\), and let \( x + x_k^+(B) \) if \( z_k^-(B) > z \) and \( z + \max[z, z_k^-(B)] \). Go to A4, (1) if \( P_k \) is solved by APPRXB1\((P_k)\) (i.e., \( z_k^+ = z_k^-(B) \)), (2) if \( z_k^+ \leq z \), or (3) if \( |D_k \cup E_k| = K - 1 \), where \( |X| \) denotes the cardinality of set \( X \). Otherwise go to A5.

A4. Let \( \mathcal{A} = \mathcal{A} \setminus \{P_k\} \) and return to A2.

A5. Select a variable \( x_{i',j'} \) satisfying
\[
(a) \text{ If } |P_k| = 0, \text{ generate three partial problems as follows:}
\]
\[
P_{k1} = P(D_k \cup \{(i', j')\}, E_k, F_k)
\]
\[
P_{k2} = P(D_k, E_k \cup \{(i', j')\}, F_k)
\]
\[
P_{k3} = P(D_k, E_k, \{(i', j')\}).
\]

Return to A2 after letting \( \mathcal{A} = \mathcal{A} \cup \{P_{k1}, P_{k2}, P_{k3}\} \setminus \{P_k\} \).

(b) \text{ If } |P_k| = 1, \text{ generate only two partial problems } P_{k1} \text{ and } P_{k2}.

Return to A2 after letting \( \mathcal{A} = \mathcal{A} \cup \{P_{k1}, P_{k2}\} \setminus \{P_k\} \).
Now let

\[(i_1, j_1), (i_2, j_2), \ldots, (i_N, j_N)\]

be the priority order in the selection of \(x^*, y^*\) in (6.5), i.e.,
\[\sigma_{i_1} \geq \sigma_{i_2} \geq \ldots \geq \sigma_{i_N}.\]

Let

\[I_k = \{(i_1, j_1), (i_2, j_2), \ldots, (i_k, j_k)\}.\]

If no \(P_k\) generated in \(\text{APPRXBK}(P, K)\) is solved by \(\text{APPRXB1}(P_k)\) or satisfies \(\bar{z}_k \leq \bar{z}\) (i.e., neither (1) nor (2) occurs in A3), all partial problems \(P_k = P(D_k, E_k, F_k, P')\) satisfying

\[D_k \cup E_k \cup F_k = I_k \text{ for some } k > 0\]
\[|D_k \cup F_k| \leq K - 1\]
\[|F_k| \leq 1\]
\[1 \leq k \leq N\]

are generated in \(\text{APPRXBK}(P, K)\). This is easily proved from the generation mechanism of \(\text{APPRXBK}\).

Next note that at least one partial problem generated from \(P_k\) in A5 contains an optimal solution of \(P_k\). Furthermore, if \(\bar{z}_k \leq \bar{z}\) holds in A3, \(P_k\) does not contain a solution which has a greater objective value than \(\bar{z}\). Therefore, if condition (3) \(|D_k \cup F_k| = K - 1\) is deleted from A3, the resulting algorithm is simply an ordinary branch-and-bound algorithm (e.g., [5, 9, 10]) using the LP values as upper bounds. Although such a branch-and-bound algorithm obtains an exact optimal solution, the computation time is likely to grow exponentially with respect to \(N\). Thus the condition \(|D_k \cup F_k| = K - 1\) in A3 is a key trick which keeps the computation time within a polynomial order, while the selection of variables by (6.5) is essential to keep \(z^{(A)}\) within a certain bound.

If \(K\) is set to 1 in \(\text{APPRXBK}(P, K)\), it halts after testing only the original problem \(P\). Thus breadth-\(K\) search with \(K=1\) is the same as breadth-1 search discussed in Section 5. Fig. 1 shows typical search trees of generated partial problems when \(K\) is set to 2 and 3. In the figure, each node represents a partial problem; the top node represents the original problem \(P\). A node \(P_k\) is placed below \(P_{k'}\) with a connecting edge if \(P_{k'}\) is generated from \(P_k\) in A5.

The next theorem summarizes the results for the computational complexity. The quality of \(z^{(A)}\) will be discussed in the next section.

**Theorem 6.1.** The computation time required for \(\text{APPRXBK}(P, K)\) is as follows.
Fig. 1 Typical search trees generated by breadth-K search. (If a node is decomposed into three nodes according to (6.6), the left one represents $P_{k_1}$, the middle one represents $P_{k_2}$, and the right one represents $P_{k_3}$. If a node is decomposed into two, the left one represents $P_{k_1}$, and the right one represents $P_{k_2}$.)
Multiple-Choice Knapsack Problems

$O(mN \log N)$ for $K = 1$,
$O(mKN^K)$ for $K > 1$.

Proof: The result for $K = 1$ was proved in Theorem 5.4. In order to consider the case of $K > 1$, first notice that the number of partial problems generated by APPRXBK($P, K$) is $O(KN^{K-1})$. This is because the number of the generated partial problem $P_k$ is bounded above by

$$3 \times \text{(the number of } P_k \text{ generated at A5 as } P_k \text{ of (6.6)})$$

since at most three partial problems are generated at each decomposition.

The latter number is not greater than the number of subsets $D_k, F_k$ with $|D_k \cup F_k| \leq K-1$ and $|F_k| \leq 1$ by (6.9), i.e., $K \cdot \frac{N}{k} \leq KN^{K-1}$. For each generated $P_k$, APPRXBK($P_k$) is executed in A3, requiring $O(mN)$ time (Theorem 5.4) if the time for sorting $\tilde{L}$ is not included. Since $\tilde{L}$ is sorted only at the beginning of APPRXBK($P, K$), the total time is given by

$$\max\{O(mN \log N), O(mN) \times O(KN^{K-1})\} = O(mKN^K).$$

7. Analysis of Breadth-$K$ Search

After a series of lemmas, we derive in this section a worst case bound of $z(A)$ calculated by APPRXBK($P, K$). It will also be shown that the obtained bound is sharp.

Lemma 7.1. If $P$ has an optimal solution with at most $(K-1)$ positive variables, then $z(A)$ obtained by APPRXBK($P, K$) satisfies $z(A) = z^0$ (the exact optimal value of $P$). On the other hand, if $P$ has no such optimal solution, APPRXBK($P, K$) generates a partial problem $P_k = P(D_k, E_k, F_k)$ such that $|D_k \cup F_k| = K-1$ and $z_k = z_0$, where $z_k$ denotes the optimal value of $P_k$.

Proof: First assume that $P$ has an optimal solution $x^0 = (x^0_{i,j})$ with at most $K-1$ positive variables. Define

$$(i, j) \in D \text{ if } x^0_{i,j} = 1$$

$$\quad (i, j) \in F \text{ if } 0 < x^0_{i,j} < 1,$$

where $|F| \leq 1$ and $|D \cup F| \leq K-1$. Assume that a $P_k = P(D, E, F)$ with some $E$ is generated in APPRXBK($P, K$), since $z^A = z^0$ has already been attained if an ancestor $P_k$, of $P_k$ in the search tree (as shown in Fig. 1) is terminated by condition (1) or (2) in A3. If $|F| \leq 1$, $z_k^{(B)}$ obtained for $P_k$ must satisfy $z_k^{(B)} = z_k^0$ (i.e., $z_k^{(B)} = z_k^0$) since all variables $x^0_{i,j}$ with $(i, j) \in D$ are already fixed to 1 in $P_k$, and $x^0_{i,j}$ is feasible in $P_k$. If $F = \{(i', j')\}$, however, assume that $Q_H$ is the last problem tested in B2 of APPRX31($P_k$). $Q_H$ is obtained from $P_k$ by
fixing certain variables to 0 but \( x_{i,j} \) is not set to 0, because each \( i \) obtained in B3 (i.e., \( i = z_{h,j} = z_{h-1,j} \)) is not equal to \( i' \) (note that \( (i', j') \) \( \in \mathcal{F} \) implies that list \( L \) contains only \( a_{i',j} \), for \( i = i' \)). Thus the LP optimal value \( z_H \) of \( Q_H \) satisfies

\[
\bar{z}_H \geq 0,
\]

but \( \bar{z}_H \) is equal to the optimal value \( z_H \) of \( Q_H \), i.e., \( \bar{z}_H = z_H \leq z^0 \) since the LP optimal solution of \( Q_H \) is feasible in \( P \) by definition of \( Q_H \). In either case, therefore, \( z^0 \) is obtained as \( z^0(B) \) for \( P_k \).

Next assume that any optimal solution of \( P \) has at least \( K \) positive variables. Take any optimal solution \( x^0 \) and let \( x^0_{i_1,j_1}, x^0_{i_2,j_2}, \ldots, x^0_{i_{k-1},j_{k-1}} \) be the first \( K-1 \) positive variables selected in the priority order (6.7). Define \( D_k \) and \( F_k \) by

\[
(i', j') \in D_k \text{ if } x^0_{i,j} = 1
\]

\[
(i', j') \in F_k \text{ if } 0 < x^0_{i,j} < 1
\]

for \( s = 1, 2, \ldots, K-1 \). We assume \( |F_k| \leq 1 \) by property (1.9). Then \( \text{APPRXBK}(P, K) \) generates \( P_k = (D_k, E_k, F_k) \) with

\[
E_k = \{ (i, j) \mid x^0_{i,j} = 0 \text{ and (i, j) has a higher priority than } (i_{K-1}, j_{K-1}) \text{ in } P_k \}
\]

as easily proved from (6.9). The optimal value \( z^0_k \) of \( P_k \) is obviously equal to \( z^0 \).

\begin{lemma}
Let \( P_k = P(D_k, E_k, F_k) \) be a partial problem with \( |D_k| = K-1 \) and \( |F_k| = 0 \) which is generated by \( \text{APPRXBK}(P, K) \). Then

\[
z^0_k(B)/\bar{z}_k = z^0_k > (4K-1)/4K.
\]

\end{lemma}

\begin{proof}
When \( \text{APPRXBL}(P_k) \) is executed in A3, assume that condition \( A' + \beta_{g+1} \geq b \) of (4.6) holds for \( Q_1, Q_2, \ldots, Q_g \) but not for \( Q_{g+1} \). As shown in the proof of Theorem 5.1,

\[
z^0_k(B)/\bar{z}_k \geq z^0_{g+1}/\bar{z}_k
\]

then follows, where \( z^0_{g+1} \) is \( z^0(T) \) (see (4.8)) for \( Q_{g+1} \). In this case, however, we have

\[
z^0_{g+1}/\bar{z}_k \geq \frac{(K-1)\theta_{g+1}}{(K-1)\theta_{g+1} + (\alpha_{g+1} - \bar{\theta})\theta_{g+1} + \theta_{g+1}} \frac{\gamma_{g+1}}{\gamma_{g+1} - 1 + \bar{\alpha}_{g+1} - \bar{\theta}}^0 \frac{\gamma_{g+1}}{\gamma_{g+1} - 1 + \bar{\alpha}_{g+1} - \bar{\theta}}^0 + \theta_{g+1} \frac{1}{1 - \phi_{g+1}}
\]

where \( \theta = \theta_{g+1} \gamma_{g+1} \). This is derived by using the same argument as in the proof of Lemma 4.2, but considering that \( K-1 \) variables \( x_{i,j} ((i, j) \in D_k) \)

Copyright © by ORSJ. Unauthorized reproduction of this article is prohibited.
are already fixed to 1 and that \( a_{i,j} \geq a_{i,j-1} \) holds for \((i, j) \in D_k\) by selection rule (6.5) in A5. Letting \( \alpha = a_{i,j} / a_{i,j-1} \) and \( \beta = a_{i,j} / a_{i,j-1} \), the right hand side of (7.5) can be rewritten as

\[
(K-1) + \frac{1-\alpha(1+\beta-\alpha)}{1-\beta}/K,
\]

where \( \alpha, \beta \) satisfy \( 1 \geq \alpha > \beta > 0 \). It is now direct to see that (7.4) takes its minimum when \( \alpha = (\beta + 1)/2 \) holds, and its value is

\[
(4K-1+\beta)/4K > (4K-1)/4K \quad \text{(since } \beta > 0) \quad \blacksquare
\]

**Lemma 7.3.** Assume that \( P \) has an optimal solution \( x^0 \) with more than \((K-1)\) positive variables \( x^0_{t, j, l} \) arranged in the order (6.7). Furthermore assume that one variable \( x^0_{t, j, l} \), with \( 1 \leq s \leq K-1 \) satisfies \( 0 < x^0_{t, j, l} < 1 \). Then

\[
z(A)/z^0 > \frac{1}{2} + \frac{1}{2\sqrt{K-1}}
\]

holds for \( K \geq 2 \).

**Proof:** Let \( P_k = P(D_k, E_k, F_k) \) be the partial problem generated in APPRXBK \((P, K)\) such that

\[
D_k = \{ (i, j, l) \mid t = 1, 2, \ldots, K-1, t \neq s \}
\]

\[
F_k = \{ (i, j, l) \}
\]

and \( E_k \) satisfies (7.3). (Such \( P_k \) is generated as shown in Lemma 7.1.) \( P_k \) satisfies \( z^0 = z^0_k \), where \( z^0_k \) is the optimal value of \( P_k \). When APPRXBl(\( P_k \)) is executed at A3, \( Q_1 = P_k \) and \( Q_{h+1} = Q_h(x^0_{t, j, l}) = 0 \) \((h = 1, 2, \ldots, H-1)\) are generated. Let \( H \) be the smallest \( h \) satisfying

\[
x^0_{t, j, l} = 1 \quad \text{for some } K \leq t \leq T.
\]

Then

\[
Q_H = P(D_H, E_H \cup E', F_H), \quad E' = \{ (i, j, l-1) \}
\]

If there is no such \( H \), the LP optimal solution of \( Q_H \) is also an optimal solution of \( Q_H \) and hence \( z(A)/z^0 \geq z(B)/z^0 \geq z_k/z^0 = z^0/z^0 = 1 \) follows. Thus assume that \( H \) exists. Obviously \( Q_H \) contains an optimal solution \( x^0 \) of \( P \), and its value (denoted \( z^0_H \)) satisfies

\[
z^0_H = \bar{z} H(x^0_T) = \bar{z} H(x^0_T).
\]

Hereafter in this proof, \( \bar{z} H \) (LP optimal solution of \( Q_H \)), \( \bar{z} H \) (the value of \( \bar{z} H \)), \( x^0_T \) \((x(T))\) obtained for \( Q_H \), \( \bar{z} H \), \( \bar{z} H \) and \( \bar{z} H \) are respectively denoted by \( \bar{z}, \bar{z}, \bar{z} \).
There exists a variable $x_{i,j}$ (say $i \neq j$) satisfying $(i, j) \in D_k$, $x_{i,j} = 1$ and $c_{i,j}^2 \geq c_{i,j} = c_{j-1}$ (this includes the case of $(i, j) = (i', j') \in F_k$). Then by an argument similar to the proofs of Lemma 4.2 and Lemma 7.2, we can derive

\[
\sum_{(i,j) \in D_k} c_{i,j}^2 + c_{i,j}^2 + c_{i,j}^2 \\
\sum_{(i,j) \in D_k} c_{i,j}^2 + c_{i,j}^2 + 6\sigma_{i,j} + (1-\theta)\sigma_{i,j}^2 - 1
\]

where $\theta = \rho_{(j-1, j)}^2 / \sigma_{i,j}^2 - 1$. Here $\sum_{(i,j) \in D_k} c_{i,j}^2 + c_{i,j}^2$ are introduced because all $x_{i,j}$, $(i, j) \in D_k$, and $x_{i,j}^2$ are fixed to 1. The term $(\sigma_{i,j}^2 - 1)\rho_{(j-1, j)}^2$ which is in (4.18) and (7.5) (it comes from $c$ discussed in (4.10) and (4.11)) disappears in (7.8) because $c_{i,j}^2$ may be included in $c$ of (4.10). Since $c_{i,j}^2 \geq c_{i,j}^2 = c_{i,j} = c_{j-1}$ for $(i, j) \in D_k$, the right hand side of (7.11) is further bounded below by

\[
(K-1)c_{i,j}^2 + c_{i,j}^2 \\
(K-1)c_{i,j}^2 + 6c_{i,j} + (1-\theta)c_{i,j}^2 - 1
\]

where $\alpha = c_{i,j}^2 / c_{i,j}^2 - 1$ and $\beta = c_{i,j}^2 / c_{i,j}^2 - 1$. $\alpha$ and $\beta$ satisfy $1 \geq \alpha > \beta > 0$. By direct calculation, it can be shown that (7.12) attains its minimum when $\beta \to 0$ and

\[
\alpha = -(K-1) + \sqrt{(K-1)^2 + (K-1)}.
\]

Substituting these into (7.12), we obtain

\[
\sum_{(i,j) \in D_k} c_{i,j}^2 + c_{i,j}^2 + 1/2, \sqrt{(K-1)/K}
\]

This proves (7.7) because $z_{i,j} \leq 1$ and $a^{(A)} \geq z_{i,j}^{(T)}$.

(b) All variables $x_{i,j}$ with $x_{i,j} = 1$ (i.e. $i \neq j$) and $(i, j) \in D_k$ satisfy

\[
c_{i,j} < c_{i,j}^2.
\]

In particular this implies $x_{i,j} = 0$ if $i ' = j$ and $j ' = i - 1$ (i.e., $x_{i,j} = c_{i,j}^2 / c_{i,j}^2$) (hence we have $x_{i',j'} = 0$ or 1), but (7.14) is not satisfied if $x_{i',j'} = 1$. In other words, $c_{i,j}^2/c_{i,j}^2$ is located to the left of $c_{i,j}^2 = c_{i,j} = c_{j-1}$. In particular, $c_{i,j}^2$ is satisfied if $x_{i,j} = x_{i,j} = 1$ and $(i, j) \not\in F_k$. Now let

\[
E = \{ (i, j) \mid (i, j) \not\in D_k \cup F_k \cup \{ (i, j) = (i, j) = 1 \} \}
\]

higher priority than $(i, j) = (i, j) = 1$ in the sense of (6.7)).
and consider the following partial problem

\[(7.16) \quad P'_k = P(D_k \cup \{(i, j-1)\}, E \cup E', P'_k) \]

\((E'\text{ is given in } (7.9)). \text{ In other words, } P'_k \text{ is } Q'_k \text{ with a restriction } x_{i,j-1} = 1 \text{ added. Note also that } E \supseteq E' \text{ holds. By (7.8), } P'_k \text{ contains an optimal solution of } P. \) We now show that \(z'_i = 0 \) still holds in the LP optimal solution \(z'\) of \(P'_k\). First note that

\[(7.17) \quad \sum_{i \neq j} a_{i,j} x_{i,j} + a_{i,j} x_{i,j-1} \geq b_k \]

holds for \(z\) (\(b_k\) is defined in (6.4)) since the computation of \(z\) for \(Q'_k\) has terminated at \(p;: (j-1, j)\) by assumption. This means that the computation of DUAL\((P'_k)\) terminates at some \(B_k\) which is located to the right of \(p;: (j-1, j)\) in list \(L\), because condition \(x_{i,j} = 1\) is further added. Thus \(x'_i = 0\) follows since \(a_{i,j} / a_{i,j}\) is located to the left of \(p;: (j-1, j)\) in \(L\).

Next construct the following problem

\[(7.18) \quad P''_k = P(D_k \cup \{(i, j-1)\}, E, F'_k) \]

by deleting \(E'\) from \(E \cup E'\). \(P''_k\) also contains an optimal solution of \(P\). The LP optimal solution \(x''\) of \(P''_k\) also satisfies \(x'' \geq 0\) since \(P''_k\) is less constrained than \(P'_k\) and hence \(\lambda'' \leq \lambda'\) holds, where \(\lambda''(x')\) is \(\lambda'\) obtained by DUAL for \(P''_k(P'_k)\).

Although \(P''_k\) is not generated in APPRXBK\((P, K)\) since \(|D_k \cup \{(i, j-1)\} \cup F'_k| = K\), the next problem is generated.

\[(7.19) \quad P^+_k = P(D_k \cup \{(i, j-1)\}, E \cup F'_k, \emptyset). \]

Now compare \(P''_k\) and \(P^+_k\). Since \(x'' = 0\) holds in the LP optimal solution of \(P''_k\), forcing \(x'' = 0\) in \(P^+_k\) does not introduce any additional restriction. Therefore we have

\[(7.20) \quad z'' = z^+, \]

where \(z''(z^+)\) is the LP optimal value of \(P''_k(P^+_k)\).

Consequently, it follows that

\[z(A)/z^0 \geq z''(z^+) \quad \text{(since } E'' \text{ contains an optimal solution of } P) \]

\[= z(A)/z^+ \quad \text{(by (7.20))} \]

\[\geq z^+(B)/z^+ \quad \text{is the value obtained by APPRXB1\((P^+_k)\))} \]

\[> (4K-1)/4K \quad \text{(by Lemma 7.2)} \]

\[> 1/2 + 1/2 \sqrt{K-1}/K. \]

This completes the proof. \(\Box\)

Theorem 7.4. The objective value \(z(A)\) obtained by APPRXBK\((P, K)\) satisfies

\[\leq \quad \text{Copyright © by ORSJ. Unauthorized reproduction of this article is prohibited.}\]
where $z_0$ is the optimal value of $P$.

Proof: Assume that $P$ has an optimal solution $x_0$ with positive variables

$$x_{i_1,j_1}^0, x_{i_2,j_2}^0, \ldots, x_{i_T,j_T}^0$$

arranged in the priority order of (6.7). If $T \leq K - 1$, $z_A = z_0$ holds as shown in Lemma 7.1. Thus assume $T > K - 1$. Next if $x_{i_1,j_1}^0 = x_{i_2,j_2}^0 = \ldots = x_{i_{K-1},j_{K-1}}^0 = 1$, consider partial problem $P_k = (D_k, F_k, F_k)$ defined by

$$D_k = \{(i', j') | t = 1, 2, \ldots, K-1\}$$

$$F_k = \emptyset$$

$$F_k = \{(i, j) | (i, j) \notin D_k \text{ and } (i, j) \text{ has a higher priority than } (i_{K-1}', j_{K-1}') \text{ in (6.7)}\}.$$ 

$P_k$ contains an optimal solution $x_k^0$ of $P$, and furthermore $P_k$ is generated in APPROXKE$(P, K)$ since $|D_k \cup F_k| = K - 1$. Then by Lemma 7.2, we have

$$z(A)/z_0 \geq z_k^0/z_k^0 \quad (z_k^0 \text{ is the optimal value of } P_k)$$

$$\geq z_k^0/z_k^0 \quad (z_k^0 \text{ is the LP optimal value of } P_k)$$

$$> (4K-1)/4K \geq 1 + \frac{1}{2} \sqrt{\frac{K-1}{K}}.$$ 

Finally if $T > K - 1$ and $x_{i,b}^0 < 1$ for some $1 \leq b \leq K - 1$, we obtain $z(A)/z_0 \geq 1 + \frac{1}{2} \sqrt{\frac{K-1}{K}}$ by lemma 7.3. 

By combining Theorem 6.1 and Theorem 7.4, the next theorem is immediate.

Theorem 7.5. For any $\epsilon$ with $0 < \epsilon < \frac{1}{4}$, an approximate solution $x(A)$ of $P$ (see (1.1)) satisfying

$$(7.22) \quad z(A)/z_0 > 1 - \epsilon$$

is obtained in $O(m \lceil \frac{1}{4(\epsilon - \epsilon^2)} \rceil N \lceil \frac{1}{4(\epsilon - \epsilon^2)} \rceil)$ time, where $z_0$ is the optimal value of $P$.

Proof: First note that

$$(7.23) \quad \frac{1}{2} + \frac{1}{2} \sqrt{\frac{K-1}{K}} \geq 1 - \epsilon \iff K \geq \frac{1}{4(\epsilon - \epsilon^2)}.$$ 

Thus breadth-search gives an approximate solution $x(A)$ satisfying $z(A)/z_0 > 1 - \epsilon$ as shown in Theorem 7.4. The result for computation time then follows from Theorem 6.1. 

In concluding this section, we show that the bound obtained in Theorem 7.4 is sharp.

Theorem 7.6. For any given $\delta > 0$ and a positive integer $K \geq 2$, there is a...
problem $P$ such that the approximate solution $x^{(A)}$ obtained by APPRXBK($P$, $K$) satisfies

$$z^{(A)}/z \leq 1 + \frac{1}{2} \sqrt{\frac{K-1}{K}} + \epsilon.$$  

**Proof:** Consider the following problem $P$ with $n = K$, $m_{i} = 2$, $i = 1, 2, \ldots, K-2$, $m_{K-1} = 1$ and $m_{K} = 2$.

$$P: a_{i1} = 1, i = 1, 2, \ldots, K-2, a_{K-1,1} = (1-\beta)/(1-\alpha), a_{K1} = 1$$

$$a_{i2} = 1/3, i = 1, 2, \ldots, K-2, a_{K2} = \beta$$

(7.24)

$$\sigma_{i1} = 1, i = 1, 2, \ldots, K$$

$$\sigma_{i2} = 1/(1 + \sqrt{2}), i = 1, 2, \ldots, K-2, \sigma_{K2} = \alpha$$

$$b = (K-2) + \alpha + (1-\beta)/(1-\alpha)$$

where

(7.25) \[ \alpha = -(K-1) + \sqrt{(K-1)^2 + (K-1)} = 1/(1 + \sqrt{K/K-1}) \] (see (7.13))

and $\beta > 0$ is an arbitrarily small positive number. $\alpha$ satisfies

(7.26) \[ 1/2 > \alpha \geq 1/(1 + \sqrt{2}). \]

Parameters relevant to performing APPRXBK($P$, $K$) are arranged below in the non-decreasing order.

$$\begin{array}{c|c|c|c|c}
\rho_{K}(1,2), a_{K-1,1}^{\sigma_{K-1,1}} & \rho_{K}^{(1,2)} & \sigma_{K1}^{\alpha_{K1}}, \sigma_{i1}^{\sigma_{i1}} & \sigma_{i2}^{\alpha_{i2}} & \sigma_{K2}^{\alpha_{K2}} \\
(i=1,2,\ldots, K-2) & (i=1,2,\ldots, K-2) & (i=1,2,\ldots, K-2) & (i=1,2,\ldots, K-2) \\
\frac{1-\alpha}{1-\beta} & \frac{3}{2}(2-\sqrt{2}) & 1 & \frac{3}{1+\sqrt{2}} & \alpha/\beta \\
\end{array}$$

(7.27)

In addition, the following tie breaking rules are adopted.

(i) When DUAL($P$) is carried out on list $L$ (which is (7.27) with $\sigma_{i1}/\sigma_{i1}$ ($i = 1, 2, \ldots, K-1$) deleted) from the right, $\sigma_{K-1,1}/\sigma_{K-1,1}$ is selected before $\rho_{K}(1, 2)$ though these have the same value.

(ii) $(i', j')$ in (6.5) of APPRXBK($P$, $K$) are selected in the following priority order, though some $\sigma_{i1}$'s assume the same value.

(7.28) \[(1,1), (2,1), \ldots, (K,1), (K,2), (1,2), (2,2), \ldots, (K-1,2). \]

Then the following LP optimal solution $\bar{x}$ and its value $\bar{z}$ of $P$ are obtained.

$$\bar{x}_{i1} = 1, i = 1, 2, \ldots, K-1, \bar{x}_{i2} = 0, i = 1, 2, \ldots, K-2$$

(7.29)

$$\bar{x}_{K1} = 1-\theta, \bar{x}_{K2} = \theta \text{ and } \theta = (1-\alpha)/(1-\beta)$$

$$\bar{z} = (K-1) + (2\alpha - \alpha^2 - \beta)/(1-\beta).$$

Copyright © by ORSJ. Unauthorized reproduction of this article is prohibited.
On the other hand, an optimal solution $x^0$ and its value $z^0$ is given by

\begin{align}
&x_{i1}^0 = 1, i = 1, 2, \ldots, K-2, K, \\
x_{i2}^0 = 0, i = 1, 2, \ldots, K-2, K
\end{align}

(7.30)

\begin{align}
x_{K-1,1}^0 &= \frac{(2a - \alpha^2 - \beta)}{(1 - \beta)} \\
z^0 &= (K-1) + \frac{(2a - \alpha^2 - \beta)}{(1-\beta)}
\end{align}

(this is obviously optimal because $z^0 = Z$).

Next consider partial problems $P_k$ generated in APPRXBK($P, K$). First note that two problems

\begin{align}
P_{k_1} &= P\left(\{(1, 1), (2, 1), \ldots, (K-1, 1)\}, \phi, \phi \right) \\
P_{k_2} &= P\left(\{(1, 1), (2, 1), \ldots, (K-2, 1)\}, \phi, \{(K-1, 1)\} \right)
\end{align}

(7.31)

are generated by the above rule (ii). These have the following approximate solution $x(B)$ and its value $z(B)$.

\begin{align}
x_{i1}^{(B)} &= \alpha, i = 1, 2, \ldots, K-1, \\
x_{i2}^{(B)} &= 0, i = 1, 2, \ldots, K-2
\end{align}

(7.32)

Note also any ancestor of $P_{k_1}$ or $P_{k_2}$ has the same $z(B)$. In all partial problems $P_k$ except for those mentioned above, at least one $x_{i'1}^{(B)}$ ($1 \leq i' \leq K-1$) is fixed to 0. Therefore the objective value $z_k(B)$ of such $P_k$ is bounded above by

\begin{align}
(K-1) + \frac{1}{1+\sqrt{2}} (\text{assuming } x_{i'1}^{(B)} = 1 (i' \neq i'), x_{i'2}^{(B)} = 1 \text{ and } x_{i'j}^{(B)} = 0) \\
&\leq (K-1) + \alpha \quad \text{(by (7.26)).}
\end{align}

(7.33)

Consequently, we have

\begin{align}
z(A)/z^0 &= (Z-K)/((K-1) + (2a - \alpha^2 - \beta)/(1-\beta)) \\
&\rightarrow \frac{(K-1) + \alpha}{(K-1) + 2\alpha - \alpha^2} = \frac{1}{2} + \frac{1}{2\sqrt{K-1/K}} \quad \text{(by (7.25)).}
\end{align}

8. Discussion

Although the multiple-choice continuous knapsack problem is NP-complete as shown in Theorem 2.3, it seems to be a rather tractable one compared with other NP-complete problems. Exact optimal solutions can be obtained quite efficiently by a branch-and-bound algorithm as reported in [6], and good approximate solutions can be obtained in polynomial time as shown in this paper.
As noted previously, the discrete 0-1 version of our problem has been studied in several papers. In particular, [3, 11] contain polynomially bounded approximate algorithms. Such algorithms may also be used as approximate algorithms for our continuous version. For a given error bound $\epsilon > 0$, the algorithm of [3] requires $O(N^{\lceil \frac{1}{\epsilon} - 1 \rceil} \log n)$ time, while the algorithm of [11] requires $O\left(\frac{nN}{\epsilon} \right)$ time. The latter is substantially faster than the former as well as than our breadth-$K$ approach, in the sense of the worst case time bound. Judging from our limited experience, however, we believe that the breadth-$K$ search method is at least competitive with the algorithm of [11] in the sense of the average computation time for the following reasons. (1) The number $H$ of the problems $Q_h$ solved in breadth-1 search applied to a partial problem $P_k$ (generated in breadth-$K$ search algorithm) is very small (typical values of $H$ observed in the experiment of [6] are less than 5 even for problems of $N \geq 1000$). (2) The number of the generated partial problems $P_k$ should also be very small by the fact that surprisingly accurate approximate value $z_k(B)$ and the upper bound $z_k$ are usually obtained for each $P_k$ (typical errors from the exact optimal value are less than 0.001% as reported in [6]); hence most $P_k$ are terminated by (1) or (2) of Step A3.

Recently, approximate algorithms for various combinatorial optimization problems receive intensive attention (e.g., [4, 7, 13, 14]). The idea of breadth-$K$ search seems to be applicable to many other problems as well, with suitable modifications incorporated.

Acknowledgment

The author wishes to thank Professor H. Mine and Professor T. Hasegawa of Kyoto University for their helpful comments.

References

Murray Hill, 1976.


Appendix A. An $O(m_i)$ algorithm to compute the set of dominant indices $J_i$.

Lemma 3.1 is proved here by actually constructing an $O(m_i)$ algorithm to compute the set of dominant indices $J_i$ for each $i$ ($1 \leq i \leq n$).

For a subset $J = \{k_1, k_2, \ldots, k_{d_i}\}$ ($k_1 < k_2 < \ldots < k_{d_i}$) of $\{1, 2, \ldots, m_i\}$, it is shown in [6] that $k_{d_i} \in J_i$ if

$$\rho_i(k_{d_i-1}, k_{d_i}) > \rho_i(k_{d_i}, k_{d_i+1})$$

holds, where $\rho_i$ is defined in (3.7). Furthermore, $J_i = \{j_1, j_2, \ldots, j_{d_i}\}$ is the set of dominant indices if it satisfies

$$\rho_i(j_1, j_2) < \rho_i(j_2, j_3) < \ldots < \rho_i(j_{d_i-1}, j_{d_i}).$$

Copyright © by ORSJ. Unauthorized reproduction of this article is prohibited.
Thus \( J_i \) can be obtained by calculating the maximum subset \( J \) which does not contain a \( k_j \) satisfying (A1). This is done by the following algorithm.

The algorithm starts with \( J = \{1, 2, \ldots, m_i\} \) and, at each iteration, eliminates from \( J \) an element \( k_j \) satisfying (A1). Let \( k_j \) always denote the \( j \)-th element from the left in \( J \) (i.e., the \( j \)-th smallest element).

Algorithm DOM(\( P, i \))

1. \( J(\{k_1, k_2, \ldots, k_m\}) = \{1, 2, \ldots, m_i\}, \ k < 2. \)
2. If \( \rho \rho(k_{j-1}, k_j) < \rho \rho(k_j, k_{j+1}) \), go to J3; else go to J4 after deleting \( k_j \) from \( J \).
3. \( J = J + 1 \). If \( j = |J| \) halt; else return to J2.
4. Let \( j = j - 1 \) if \( j - 1 = 1 \); do not change \( j \) if \( j - 1 = 1 \). If \( j = |J| \) (this is possible only when \( j \) does not change), halt; else return to J2.

Upon termination in J3 or J4, \( J_i = J \) holds. The validity of the algorithm may be obvious from the comments given above. The time complexity is considered below.

Since each step J1-J4 requires a constant time, the complexity is measured by counting how many times J3 or J4 is visited. Let

\[
(A3) \quad p = |J_i| - 1.
\]

Initially \( p \) is \( m_i - 2 \). \( p \) decreases by one at J3 always, and at J4 if \( j - 1 = 1 \) holds. If \( j = j + 1 \) - 1 if \( j = j - 1 \) holds. Consequently either \( p \) or \( |J| \) decreases by one whenever J3 or J4 is visited. Since \( p = m_i - 2 \) and \( |J| = m_i \) initially, and \( p > 0 \) and \( |J| > 1 \) must hold, the above argument proves that J3 and J4 are executed \( 2m_i - 4 \) times in total. Thus the time complexity of DOM(\( P, i \)) is \( O(m_i) \).

Appendix B. Proof of Lemma 5.3.

For notational simplicity, assume that list

\[
(B1) \quad \tilde{J} = \{\tilde{j}_1, \tilde{j}_2, \ldots, \tilde{j}_{d-1}, \tilde{j}_d\}, \quad \tilde{a}_1, \tilde{a}_d \}
\]

is given for a set of dominant indices

\[
(B2) \quad J_i = \{j_1, j_2, \ldots, j_d\} \text{ of } J = \{1, 2, \ldots, m_i\}.
\]

Assume further that

\[
(B3) \quad \tilde{r} = \rho \rho_j, \tilde{j}_k+1), \text{ where } 1 \leq \tilde{r} \leq d - 1,
\]

holds and \( \tilde{j}_k \) is deleted from \( J \) to obtain \( J' = \{1, 2, \ldots, j_{k-1}, j_{k+1}, \ldots, m_i\} \).
Let $L'_i$ be obtained from $J'_i$ (the set of dominant indices of $J'$) in the same manner as $L_i$ for $J_i$. We show that any $\beta$ which is in exactly one of $L_i$ and $L'_i$ satisfies

\[
\beta \leq \bar{\lambda}.
\]

Before proceeding to the proof, construct a graph $G_i$ to visualize the computation process of $J_i$ (and hence $L_i$). It starts with the following set of nodes labeled with pairs of indices,

\[
(1, 2), (2, 3), \ldots, (m_i - 1, m_i).
\]

All nodes are defined to be uncovered. The following operation MERGE is then repeated as long as possible.

**MERGE**: Take any two uncovered nodes $(p, q)$ and $(q, r)$ ($p < q < r$) such that $\rho_i(p, q) \leq \rho_i(q, r)$, and form a new node $(p, r)$. $(p, r)$ is placed above $(p, q)$ and $(q, r)$, and edges $((p, r), (p, q))$ and $((p, r), (q, r))$ are drawn. $(p, q)$ and $(q, r)$ are now covered and $(p, r)$ is uncovered.

Fig. Bl(a) illustrates an example of $G_i$ for $J_i = \{1, 2, \ldots, 12\}$. The obtained set of uncovered nodes is $\{(1, 3), (3, 7), (7, 12)\}$. The following properties may be obvious from the definition of $G_i$.

(a) Let the resulting set of uncovered nodes in $G_i$ be $\{(j_1, j_2), (j_2, j_3), \ldots, (j_{d-1}, j_d)\}$. Then $J_i = \{j_1, j_2, \ldots, j_d\}$ and $L_i = \{\rho_i(j_1, j_2), \ldots, \rho_i(j_{d-1}, j_d)\}$ holds, where $\rho_i(j_1, j_2) < \rho_i(j_2, j_3) < \rho_i(j_{d-1}, j_d) < \cdots$. 

(b) $G_i$ is not unique, but any $G_i$ has the same set of uncovered nodes.

(c) Any index $j_k (\neq 1, m_i) \in J_i$ appears in two uncovered nodes $(p, j_k)$ and $(j_k, q)$, where $p < j_k$ and $j_k < q$, while $j_k = 1$ or $j_k = m_i$ always appears in exactly one uncovered node.

(d) Let $(p, r)$ be generated from $(p, q)$ and $(q, r)$ by MERGE. Then $\rho_i(p, q) \geq \rho_i(p, r) \geq \rho_i(q, r)$ holds as directly proved from the following properties

\[
\begin{align*}
\frac{b}{a} > \frac{d}{c} & \quad (a, b, c, d > 0) \quad \text{implies} \quad \frac{b + d}{a + c} > \frac{d}{c} \\
\frac{b}{a} > \frac{d}{c} & \quad (a, b, c, d > 0) \quad \text{implies} \quad \frac{b + d}{a + c} = \frac{d}{c}.
\end{align*}
\]

Now assume that $j_k (\neq m_i) \in J_i$ is deleted from $J$ to obtain $J'$. $G'_i$ (i.e., $G_i$ for $J'$) is obtained by

(1) deleting from $G_i$ all the nodes containing $j_k$ and also all the edges connected to them,

(2) adding uncovered node $(j_{k-1}, j_{k+1})$ (if $j_k = 1$, this is not necessary).
Multiple-choice Knapsack Problems

Fig. B1. Example of $G_i$ and the computation of $G'_i$. 

Copyright © by ORSJ. Unauthorized reproduction of this article is prohibited.
and

(3) applying MERGE as long as possible to the resulting graph.

Fig. 1B(b)(c) exemplifies this process.

Among all possible $G_i$ (see comment (b) above), we assume here that $G_i$ is obtained as follows.

(i) First $G_i^1$ and $G_i^2$ for $J = \{1, 2, \ldots, j_{k-1}\}$ and $J = \{j_{k+1}, \ldots, m_i\}$ are respectively obtained ($J^1 = \emptyset$ in case of $j_k = 1$).

(ii) $G_i$ is then obtained from the union of $G_i^1$ and $G_i^2$ with two nodes $(j_{k-1}, j_k)$ and $(j_k, j_{k+1})$ added.

Under these assumptions, it is easy to see that the graph obtained after (1) above is exactly the union of $G_i^1$ and $G_i^2$. Thus $G_i^1$ is obtained by adding one node $(j_{k-1}, j_{k+1})$ to $G_i^1$ instead of adding two nodes $(j_{k-1}, j_k)$ and $(j_k, j_{k+1})$.

Now we are ready to prove Lemma 5.3. First note that the elements which are in $L_i$ but not in $L_{i-1}$ are only $\rho_i(j_{k-1}, j_k)$ and $\rho_i(j_k, j_{k+1})$. These satisfy

$$\rho_i(j_{k-1}, j_k) < \rho_i(j_k, j_{k+1}) = \lambda$$

Next we consider the elements which are in $L_i$ but not in $L_{i-1}$, i.e., these uncovered nodes newly generated in steps (1)-(3) above. Now let $(t, j_{k+1})$ be the node in $G_i$ with the smallest index $t$ satisfying $t > j_k$. The position of $(t, j_{k+1})$ is schematically illustrated in Fig. B2. After applying (1)-(3), two cases are possible.

(A) Node $(t, j_{k+1})$ remains uncovered (note that $(t, j_{k+1})$ is an uncovered node in $G_i^1 \cup G_i^2$ after (1) is executed): Then any new uncovered node $(u, v)$ is located to the left of $(t, j_{k+1})$ in $G_i^1$. This means

$$\rho_i(u, v) < \rho_i(t, j_{k+1})$$

by definition of $G_i^1$. However

$$\rho_i(t, j_{k+1}) \leq \rho_i(j_k, j_{k+1}) = \lambda$$

holds by property (d) mentioned above.

(B) Node $(t, j_{k+1})$ becomes covered: Let $(s, j_{k+1})$ be the node in $G_i$ with the smallest index $s$. The generation process of $(s, j_{k+1})$ is illustrated in Fig. B2 by broken edges. Now note that

$$\rho_i(j_{k-1}, j_k) < \rho_i(j_{k-1}, j_{k+1})$$

is directly derived from

$$\rho_i(j_{k-1}, j_k) < \rho_i(j_{k-1}, j_{k+1}) < \rho_i(j_{k}, j_{k+1}) < \rho_i(j_{k}, j_{k+1}) < \rho_i(j_{k}, j_{k+1})$$

(by properties (a) and (d)).
Fig. B2. Position of \((t, j_{k+1})\) and generation of \((s, j_{k+1})\)
(broken edges indicate the newly generated part).
and property (B5). (B8) and the fact that \((s, j_{k+1})\) is a node in \(G_i^1\) imply that \((s, j_k)\) is a node in \(G_i^2\), as easily proved by the similarity between the construction of \(G_i^2\) (i.e., (1) and (ii)) and the construction of \(G_i^1\) (i.e., (1)-(3)). Then we have

\[ \rho_i^x(s, j_k) = \rho_i^x(j_k, j_{k+1}) \]

by applying properties (a) and (d). However this means

(B9) \[ \rho_i^x(s, j_k) < \rho_i^x(s, j_{k+1}) < \rho_i^x(j_k, j_{k+1}) = \bar{\lambda} \]

by (B5). In addition, there is no new uncovered node in \(G_i^1\) to the right of \((s, j_{k+1})\). This is because operation MERGE cannot be applied between \((s, j_{k+1})\) and an uncovered node \((u, v)\) to the right of \((s, j_{k+1})\) in \(G_i^1 \cup G_i^2\) since

\[ \rho_i^x(s, j_{k+1}) < \rho_i^x(j_k, j_{k+1}) < \rho_i^x(u, v) \]

holds. This also proves that \((s, j_{k+1})\) is an uncovered node in \(G_i^1\). Finally any new uncovered node \((u, v)\) which is located to the left of \((s, j_{k+1})\) in \(G_i^1\) satisfies

(B10) \[ \rho_i^x(u, v) < \rho_i^x(s, j_{k+1}) < \bar{\lambda} \]

(see property (a)). From (B7) (B9) and (B10), Lemma 5.3 is proved. \(\square\)

Toshihide IBARAKI: Department of
Applied Mathematics and Physics,
Faculty of Engineering,
Kyoto University,
Kyoto 606, Japan

Copyright © by ORSJ. Unauthorized reproduction of this article is prohibited.