Journal of the Operations Research
Society of Japan
Vol. 24, No. 4, December 1981

AN ALGORITHM FOR SOLVING BILINEAR KNAPSACK PROBLEMS

Hiroshi Konno
University of Tsukuba

(Received May 9, 1981)

Abstract This paper proposes a cutting plane algorithm for solving 0-1 bilinear knapsack problem (BK), a special kind of integer quadratic programming problem in which a bilinear function $c^tx + d^ty + x^tCy$ is to be maximized subject to disjoint knapsack constraints on x and y. This problem has applications in bipartite matching, multi-attribute utility analysis, cutting stock problems, and so on.

The main purpose of this paper is to demonstrate the practicality of cutting plane algorithm which is an elaboration of similar algorithms proposed for bilinear linear programming problems and convex maximization problems. It will be shown that a very effective cut can be generated without excessive amount of computation and that our approach is potentially advantageous over the standard linearization approach.

1. Introduction

This paper proposes a finitely convergent cutting plane algorithm for solving 0-1 bilinear knapsack problem (BK), a special type of 0-1 integer quadratic programming problem to be defined below:

$$\text{(BK)}\quad \begin{cases} \text{maximize} & \phi(x, y) = \sum_{i=1}^{m} c_i x_i + \sum_{j=1}^{n} d_j y_j + \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij} x_i y_j \\ \text{subject to} & \sum_{i=1}^{m} a_i x_i \le a_0, \quad x_i \in \{0, 1\}, \quad i=1, \ldots, m, \\ & \sum_{j=1}^{n} b_j y_j \le b_0, \quad y_j \in \{0, 1\}, \quad j=1 \ \ldots, n. \end{cases}$$

This problem has applications in bipartite matching problems, cutting stock problems, multi-attribute utility analysis to name only a few. BK is the simplest discrete analogue of the bilinear linear programming problem (BL):

$$(\mathrm{BL}) \begin{cases} \text{maximize} & \sum_{i=1}^{m} c_i x_i + \sum_{j=1}^{n} d_j y_j + \sum_{i=1}^{m}\sum_{j=1}^{n} e_{ij} x_i y_j \\ \text{subject to} & \sum_{i=1}^{m} a_{ri} x_i \le a_r, \quad r=1, \ldots, k; \quad x_i \ge 0, \quad i=1, \ldots, m, \\ & \sum_{j=1}^{n} b_{sj} y_j \le b_s, \quad s=1, \ldots, l; \quad y_j \ge 0, \quad j=1, \ldots, n. \end{cases}$$

which has attracted more attention in recent years. For example, the present author proposed a cutting plane algorithm [8] and showed that it pertains to many real world applications [9]. Readers are referred to [3, 4, 8, 13, 15] for algorithms and to [2, 9, 12, 14] for applications of BL.

It is true that BK can be reformulated as a standard 0-1 integer linear program by introducing new variables and constraints [6]. But this manipulation usually increases the size of the problem and destroys the problem structure.

The algorithm to be developed in this paper, on the other hand, directly exploits the special structure of the problem like the ones proposed by the present author for BL [8] and for convex maximization problem [10]. This algorithm consists of two subprocedures. One is to obtain a local maximum which amounts to solving a sequence of 0-1 knapsack problems and small scale 0-1 integer linear programs. The other is to adjoin a cutting plane which eliminates a local maximum and yet does not eliminate any solution potentially better than the current incumbent. It will be shown in Section 3 that cut coefficients can be obtained by solving parametric knapsack problems (or parametric linear programming problems). Also, finite convergence of the algorithm is established without using expensive cuts such as disjunctive face cut [12].

Though this algorithm can be easily extended to general 0-1 bilinear integer programs (e.g., bilinear assignment problem), we will concentrate here on BK because (i) it is one of the most important problems in this class and (ii) its structure pertains to various algorithmic elaboration. Numerical results on this algorithm for the problems of the size up to m=20, n=100 will be reported in Section 4. All the test problems were solved successfully within a reasonable amount of time.

2. Alternate Mountain Climbing to Obtain a Local Star Maximum

Consider a 0-1 bilinear knapsack problem:

$$(2.1)\qquad \begin{cases} \text{maximize} & \phi(x, y) = \sum_{i=1}^{m} c_i x_i + \sum_{j=1}^{n} d_j y_j + \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij} x_i y_j \\ \text{subject to} & x \in X_0, \quad y \in Y_0. \end{cases}$$

where

$$(2.2)\qquad X_0 = \{x \in R^m \mid \sum_{i=1}^{m} a_i x_i \leq a_0, \quad x_i \in \{0, 1\}, \quad i=1, \ldots, m\}$$

$$(2.3)\qquad Y_0 = \{y \in R^n \mid \sum_{j=1}^{n} b_j y_j \leq b_0, \quad y_j \in \{0, 1\}, \quad j=1 \ \ldots, n\}$$

It will be assumed throughout that a_i's and b_j's are positive integers and that c_i's, d_j's and c_{ij}'s are (not necessarily positive) integers. Also, we will assume that $X_0 \neq \phi$, $Y_0 \neq \phi$ and that (x^*, y^*) is the current incumbent, i.e., the best feasible solution of (2.1) identified to date.

A natural step to obtain a local maximum is to maximize ϕ by alternately fixing the value of x or y, which amounts to solving a sequence of knapsack problems. To simplify the notation, let

$$(2.4)\qquad K_X(y): \text{maximize } \{\phi(x, y) \mid x \in X\}$$

$$(2.5)\qquad K_{Y_0}(x): \text{maximize } \{\phi(x \ \ y) \mid y \in Y_0\}$$

The set $X \subset X_0$ is initially taken to be X_0, in which cases $K_{X_0}(y)$ as well as $K_{Y_0}(x)$ is a 0-1 knapsack problem. Note, however, that constraints will be added to X_0 in the later steps of the algorithm, so that $K_X(y)$ will, in general, be a 0-1 integer program without special structures.

Procedure AMC(X, Y_0; x^0, y^0) (Alternate Mountain Climbing)

Step 0. Given $x^0 \in X_0$ and $y^0 \in Y_0$, let $k=1$.

Step 1. Obtain an optimal solution x^k of $K_X(y^{k-1})$ and an optimal solution y^k of $K_{Y_0}(x^k)$.

Step 2. If $\phi(x^k, y^k) > \phi(x^{k-1}, y^{k-1})$, then let $k=k+1$ and go to Step 1. Otherwise, let $(\tilde{x}, \tilde{y}) = (x^k, y^k)$ and go to Step 3.

Step 3. If $\phi(\tilde{x}, \tilde{y}) > \phi(x^*, y^*)$, then let $(x^*, y^*) = (\tilde{x}, \tilde{y})$ and halt.

The pair of points, $(\tilde{x}, \tilde{y})$ defined above will be called a locally maximal pair.

Theorem 1. Procedure AMC(X, Y_0; y^0) generates a locally maximal pair in finitely many steps.

Proof: Follows from the finiteness of X and Y_0 and from the monotonically increasing property of the sequence $\phi(x^k, y^k)$. ||

Once a locally maximal pair is reached, one cannot improve the objective function by fixing either x equal to $\tilde{x}$ or y equal to $\tilde{y}$. In this case, one will switch to a semi-global optimization process.

Let $x^{(i)}$ be the i-th complement of $x \in X_0$, i.e.,

$$x^{(i)} = (x_1, \ldots, x_{i-1}, 1-x_i, x_{i+1}, \ldots, x_m) \tag{2.6}$$

and let

$$I(x) = \{i \mid x^{(i)} \in X\}. \tag{2.7}$$

Definition. $(\hat{x}, \hat{y})$ is a local star maximum if $(\hat{x}, \hat{y})$ is a locally maximal pair and

$$\max \{\phi(\hat{x}^{(i)}, y) \mid y \in Y_0\} \leq \phi(\hat{x}, \hat{y}) \tag{2.8}$$

holds for all $i \in I(\hat{x})$. ||

It is easy to see that the procedure defined below generates a local star maximum in finitely many steps.

Procedure SGO (X, Y_0) (Semi-Global Optimization)

Step 0. Choose $y^0 \in Y_0$ arbitrarily.

Step 1. Execute AMC(X, Y_0: x^0, y^0) and let $(\hat{x}, \hat{y})$ be a locally maximal pair.

Step 2. If $\max \{\phi(\hat{x}^{(i)}, y) \mid y \in Y_0\} \leq \phi(\hat{x}, \hat{y})$, $\forall i \in I(\hat{x})$, then halt. Otherwise, go to Step 3.

Step 3. Let $y^0 = \bar{y}$ where $\bar{y} \in Y_0$ satisfies $\phi(\hat{x}^{(i)}, \bar{y}) > \phi(\hat{x}, \hat{y})$ for some $i \in I(\hat{x})$. Go to Step 1.

3. Cutting Planes from a Local Star Maximum

Given a local star maximum $(\hat{x}, \hat{y})$, the next step is to introduce a cutting plane which eliminates $\hat{x}$ and yet does not eliminate any $x \in X_0$ for which

$$\max \{\phi(x, y) \mid y \in Y_0\} > \phi(x^*, y^*) \tag{3.1}$$

Such a cutting plane will be called "valid". Note that (3.1) is tantamount to

$$\max \{\phi(x, y) \mid y \in Y_0\} \geq \phi(x^*, y^*) + 1 \tag{3.2}$$

since c_i's, d_j's, c_{ij}'s as well as x_i's and y_j's are integers.

For simplicity, let

$$z_i = \begin{cases} x_i, & \text{if } \hat{x}_i = 0 \\ 1 - x_i, & \text{if } \hat{x}_i = 1 \end{cases} \qquad i = 1, \ldots, m \tag{3.3}$$

so that $z = 0$ corresponds to $\hat{x}$.

Let $\psi(z, y)$ be the expression of $\phi(x, y)$ relative to a new set of variables, i.e.,

$$\begin{aligned} \phi(x, y) &= \sum_{i=1}^{m} c_i x_i + \sum_{j=1}^{n} d_j y_j + \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij} x_i y_j \\ &= \sum_{i=1}^{m} \gamma_i z_i + \sum_{j=1}^{n} \delta_j y_j + \sum_{i=1}^{m}\sum_{j=1}^{n} \gamma_{ij} z_i y_j + \phi(\hat{x}, 0) \\ &= \psi(z, y) \end{aligned} \tag{3.4}$$

Also, let

$$g_i(\lambda) = \max \{\psi(z, y) \mid 0 \leq z \leq \lambda e^i, \quad y \in Y_0\}, \quad i = 1, \ldots, m \tag{3.5}$$

where $\lambda \geq 0$ and e^i represents the i-th unit vector. Note that

$$g_i(0) = \phi(\hat{x}, \hat{y}) \tag{3.6}$$

since by definition,

$$g_i(0) = \max \{\psi(0, y) \mid y \in Y_0\} = \max \{\phi(\hat{x}, y) \mid y \in Y_0\} = \phi(\hat{x}, \hat{y})$$

Lemma 2: g_i is convex on $[0, \infty)$ for all i.

Proof: The maximum of the right hand side of (3.5) is attained either at $z = 0$ or at $z = \lambda e^i$ since ψ is linear in z. Hence we have by (3.6)

$$g_i(\lambda) = \max [\phi(\hat{x}, \hat{y}), \quad h_i(\lambda)] \tag{3.7}$$

where

$$h_i(\lambda) = \max \{\sum_{j=1}^{n} (\delta_j + \gamma_{ij}\lambda) y_j \mid y \in Y_0\} + \gamma_i \lambda + \phi(\hat{x}, 0) \tag{3.8}$$

It is straightforward to see that h_i is convex, hence g_i is convex via (3.7)||

Given g_i, let

$$\hat{\lambda}_i = \max \{\lambda \mid g_i(\mu) \leq \phi(x^*, y^*) + 1, \quad \forall\, \mu \in [0, \lambda]\} \tag{3.9}$$

Lemma 3: $\hat{\lambda}_i > 0$ for all i. Also $\hat{\lambda}_i \geq 1$ for all i such that $\hat{x}^{(i)} \in X_0$.

Proof: $\hat{\lambda}_i > 0$ follows from the definition (3.9) by noting the relation (3.6) and convexity of g_i. If $\hat{x}^{(i)} \in X$, then

$$h_i(1) = \max \{\phi(\hat{x}^{(i)}, y) \mid y \in Y_0\} \leq \phi(\hat{x}, \hat{y})$$

since $(\hat{x}, \hat{y})$ is a local star maximum. Also if $\hat{x}^{(i)} \in X_0 - X$, then

$$h_i(1) = \max \{\phi(\hat{x}^{(i)}, y) \mid y \in Y_0\} \leq \phi(x^*, y^*) + 1$$

since $\hat{x}^{(i)}$ has been cut off by a valid cut adjoined earlier. Therefore,

$$g_i(1) = \max \{\phi(\hat{x}, \hat{y}), h_i(1)\} \leq \phi(x^*, y^*) + 1$$

for all i such that $\hat{x}^{(i)} \in X_0$. The lemma follows from this by noting (3.6) and the convexity of g_i. ||

Let us define mutually exclusive sets of indices:

$$I = \{i \mid \hat{\lambda}_i < \infty\} \tag{3.10}$$

$$J = \{i \mid \hat{\lambda}_i = \infty\} \tag{3.11}$$

For $i \in J$, let

$$G_i(\mu) = \min \{\psi(z, y) \mid -\mu e^i \leq z \leq 0; \quad y \in Y_0\} \tag{3.12}$$

$$\hat{\mu}_i = \max \{\mu \mid G_i(\mu) \leq \phi(x^*, y^*) + 1\} \tag{3.13}$$

Lemma 4: $\hat{\mu}_i > 0$ for all $i \in J$.

Proof: Follows from the continuity of G_i and from $G_i(0) = \phi(\hat{x}, \hat{y}) \leq \phi(x^*, y^*) + 1$. ||

Theorem 5:

$$\sum_{i \in I} z_i/\hat{\lambda}_i - \sum_{i \in J} z_i/\hat{\mu}_i \geq 1 \tag{3.14}$$

is a valid cut.

Proof: This can be proved by applying a general theorem established in [7], but we will give an elementary proof for completeness. Let

$$Z = \{z \in R^m \mid \sum_{i \in I} z_i/\hat{\lambda}_i - \sum_{i \in J} z_i/\hat{\mu}_i \leq 1, \; z_i \geq 0, \; i = 1, \ldots, m\} \tag{3.15}$$

One needs to show

$$\max \{\psi(z, y) \mid y \in Y_0\} \leq \phi(x^*, y^*) + 1, \quad \forall z \in Z \tag{3.16}$$

It is easy to see that the extreme points of Z are given by

$$z^i = \hat{\lambda}_i e^i, \quad i \in I$$

and the extreme directions of Z are given by

(i) $e^j, \quad j \in J.$

(ii) $\hat{\mu}_j e^j + \hat{\lambda}_i e^i, \quad j \in J, \quad i \in I.$

where e^i and e^j are the i-th and j-th unit vector, respectively. Hence $z \in Z$ can be expressed as

$$z = \sum_{i \in I} \theta_i z^i + \sum_{j \in J} \alpha_j e^j + \sum_{i \in I} \sum_{j \in J} \alpha_{ij} (\hat{\mu}_j e^j + \hat{\lambda}_i e^i)$$

where

$$\sum_{i \in I} \theta_i = 1, \quad \theta_i \geq 0, \ \forall\, i \in I; \quad \alpha_j \geq 0, \ \forall\, j \in J; \quad \alpha_{ij} \geq 0, \quad \forall\, i \in I, \quad \forall\, j \in J$$

so that

$$\psi(z, y) = \sum_{i \in I} \theta_i \psi(z^i, y) + \sum_{j \in J} \alpha_j \psi(e^j, y) + \sum_{i \in I} \sum_{j \in J} \alpha_{ij} \psi(\hat{\mu}_j e^j + \hat{\lambda}_i e^i, y)$$

Hence

$$\max \{\psi(z, y) \mid y \in Y_0\} \leq \sum_{i \in I} \theta_i \max\{\psi(z^i, y) \mid y \in Y_0\} + \sum_{j \in J} \alpha_j \max\{\psi(e^j, y) \mid y \in Y_0\}$$
$$+ \sum_{i \in I} \sum_{j \in J} \alpha_{ij} \max\{\psi(\hat{\mu}_j e^j + \hat{\lambda}_i e^i, y) \mid y \in Y_0\}$$

The first term of the right hand side is less than $\phi(x^*, y^*) + 1$ since $\max \{\psi(z^i, y) \mid y \in Y_0\} \leq \phi(x^*, y^*) + 1, \forall\, i \in I$, since $(\hat{x}, \hat{y})$ is a local star maximum.

Also, $\max \{\psi(e^j, y) \mid y \in Y_0\} \leq 0, \forall\, j \in J.$ (Note $\hat{\lambda}_j = \infty$). In addition,

$$\begin{aligned}
&\max \{\psi(\hat{\lambda}_i e^i + \hat{\mu}_j e^j, y) \mid y \in Y_0\} \\
&= \max \{\psi(\hat{\lambda}_i e^i, y) - \psi(-\hat{\mu}_j e^i, y) \mid y \in Y_0\} \\
&\leq \max \{\psi(\hat{\lambda}_i e^i, y) \mid y \in Y_0\} - \min \{\psi(-\hat{\mu}_j e^j, y) \mid y \in Y_0\} \\
&\leq \{\phi(x^*, y^*) + 1\} - \{\phi(x^*, y^*) + 1\} \\
&= 0
\end{aligned}$$

This establishes (3.16). ||

Corollary 6:

(3.17) $$\sum_{i\in I} z_i/\hat{\lambda}_i \geq 1$$

is a valid cut.

Proof: Let $z\in R^m$ be a 0-1 point which is potentially better than the current incumbent. By theorem 5, we have

$$\sum_{i\in I} z_i/\hat{\lambda}_i \geq 1 + \sum_{j\in J} z_j/\hat{\mu}_j \geq 1$$

which implies that (3.17) is valid. ||

For those i's for which $x^{(i)} \notin X_0$, $\hat{\lambda}_i$ can be strictly less than 1. However, a very simple geometric observation leads us to the following theorem:

Theorem 7: Let

(3.18) $$\bar{\lambda}_i = \max \{1, \hat{\lambda}_i\}$$

then

(3.19) $$\sum_{i\in I} z_i/\bar{\lambda}_i \geq 1.$$

is a valid cut.

Proof: It suffices to show that all 0-1 points satisfying $\sum_{i\in I} z_i/\hat{\lambda}_i \geq 1$ also satisfies $\sum_{i\in I} z_i/\bar{\lambda}_i \geq 1$. Let

$$L = \{i \mid \hat{\lambda}_i < 1\} \subset I$$

and assume that there exists a 0-1 vector z such that $\sum_{i\in I} z_i/\hat{\lambda}_i \geq 1$ and $\sum_{i\in I} z_i/\bar{\lambda}_i < 1$. Then

$$\sum_{i\in L} z_i/\hat{\lambda}_i + \sum_{i\in I-L} z_i/\hat{\lambda}_i \geq 1$$

$$\sum_{i\in L} z_i + \sum_{i\in I-L} z_i/\hat{\lambda}_i < 1$$

which implies that there exists at least one $i\in L$ for which $z_i \neq 0$, i.e., $z_i = 1$.

This in turn implies that $\sum_{i\in I-L} z_i/\hat{\lambda}_i < 0$, a contradiction. ||

Remark: $\sum_{i\in I} z_i/\bar{\lambda}_i - \sum_{i\in J} z_i/\hat{\mu}_i \geq 1$ may not be a valid cut since $\max \{\psi(\bar{\lambda}_i e^i, y) \mid y\in Y_0\}$ need not be less than or equal to $\phi(x^*, y^*) + 1$.

Now we are ready to present the cutting plane algorithm for solving 0-1 bilinear knapsack problems:

Cutting Plane Algorithm CBK(X_0, Y_0)

Step 0. Let $X = X_0$.

Step 1. Execute SGO (X, Y_0).

Step 2. Compute $\hat{\lambda}_i$'s and $\hat{\mu}_j$'s for $i\in I$ and $j\in J$.

(3.20) $$X := X \cap \{z \mid \sum_{i\in I} z_i/\hat{\lambda}_i - \sum_{j\in J} z_j/\hat{\mu}_i \geq 1\}$$

Step 3. If $X\neq\phi$ then go to Step 1. Otherwise, stop ((x^*, y^*) is an optimal solution of (2.1)).

Theorem 9. CBK(X_0, Y_0) generates an optimal solution of 0-1 bilinear knapsack problems in finitely many steps.

Proof: At least one integral point of X is eliminated every time a new cut is adjoined. Hence X will become empty after finitely many cuts are added to X_0. ||

Several comments are in order. First, formula (3.20) may be replaced by:

(3.21) $$X := X \cap \{z \mid \sum_{i\in I} z_i/\bar{\lambda}_i \geq 1\}$$

without invalidating Theorem 9. Second, we would have a deeper cut (i.e., $\hat{\lambda}_i$'s and $1/\hat{\mu}_j$'s are larger) if we have a better incumbent (x^*, y^*). Hence it would be advisable to execute procedure AMC(X_0, Y_0; x^0, y^0) by taking several randomly chosen starting points x^0, y^0, prior to the start of CBK(X_0, Y_0). Many efficient algorithms have been proposed for 0-1 knapsack problems and this pre-computation would certainly be paid off. Third, we need to solve parametric knapsack problems (3.8), (3.12) to compute the exact value of $\hat{\lambda}_i$'s and $\hat{\mu}_j$'s. However, several efficient approximation procedures are available, two of which will be described in the appendix. Fourh, the algorithm developed in this paper can in principle be adapted to general bilinear programming problems with 0-1 variables:

(3.22)
$$\begin{cases} \text{maximize} & \sum_{i=1}^{m} c_i x_i + \sum_{j=1}^{n} d_j y_j + \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij} x_i y_j \\ \text{subject to} & \sum_{i=1}^{m} a_{ri} x_i \leq a_{ro}, \quad r = 1, \ldots, p; \\ & \sum_{j=1}^{n} b_{sj} y_j \leq b_{so}, \quad s = 1, \ldots, q; \end{cases}$$

$$x_i \in \{0, 1\}, \quad i = 1, \ldots, m;$$
$$y_j \in \{0, 1\}, \quad j = 1, \ldots, n.$$

Typically our algorithm is expected to work more efficiently on this problem when m is relatively small ($m \leq 30$) and x is tightly constrained. It is also desirable that constraints in y-space to have some nice structure (e.g. assignment constraints) so that subproblem in y-space can be solved efficiently.

4. Results of Numerical Experiments

We solved about eighty BK's using CBK(X_0, Y_0), in which formula (3.20) was replaced by (3.21). Standard DP algorithm was used to solve $K_{Y_0}(x)$, whereas a modified version of RIP-30C code [5] was used to solve $K_X(y)$. We applied procedure AMC(X_0, Y_0; x^0, y^0) ten times using ten randomly generated starting solutions, prior to the execution of CBK(X_0, Y_0). Also, discrete search scheme given in appendix was used to obtain cut coefficients. Program was coded in FORTRAN IV and computation was done on HITAC M/170. The main purpose of this experiment was (i) to demonstrate the practicality of our approach and (ii) to count the number of cuts to be added before termination to check the effectiveness of cutting planes.

Tables 1 and 2 list some of the results. The density of nonzero coefficients c_{ij}'s were in the range of 20 to 50 percent. It turned out that optimal solutions have been found for more than 90% of our test problems by applying procedure AMC(X_0, Y_0; y^0) ten times before the start of CBK(X_0, Y_0). Table 3 shows the statistics for 59 test problems of size (m, n) = (20, 40). Total CPU time shows a large standard deviation as is the case for most integer programming problems. Standard deviation of the number of added cuts, on the other hand, is rather small. 42 out of 59 test problems were solved by adding no more than 3 cuts. This shows that our cut is usually very deep. Another code using more efficient branch and bound algorithm for solving $K_{Y_0}(x)$ ([1, 11]) and Newton-type procedure to obtain cut coefficient [13] is now under development. Our tentative target is to solve problems of size m=40. The results of experiments using this code will be reported elsewhere.

Table 1 $m = 10$

n	number of added cuts	number of times AMC was applied	total CPU time (sec)
10	1	3	2
20	3	126	5
30	1	5	4
40	2	40	8
50	2	33	11
60	2	41	14
70	3	52	30
100	3	61	41

Table 2 $m = 20$

n	number of added cuts	number of times AMC was applied	total CPU time (sec)
20	2	623	9
30	1	57	19
40	4	1152	55
50	6	46875	584
60	6	16102	280
70	8	16842	389
100	7	24212	451

Also Table 3 shows the statistics for 59 randomly generated 20 × 40 test problems.

Table 3

	number of added cuts	total CPU time (sec)
Average	2.9	64.4
Minimum	1	7
Maximum	9	265
Standard deviations	1.6	64.6

Appendix

Approximate Procedures to Obtain Coefficients of a Valid Cut.

One has to compute $h_i(\lambda)$ to obtain an exact value of cut coefficients $\hat{\lambda}_i$ in (3.9), which amounts to solving a parametric knapsack problem:

$$\text{maximize } \{\sum_{j=1}^{n}(\delta_j + \lambda\gamma_{ij})y_j \mid \sum_{j=1}^{n} b_j y_j \leq b_0, \quad y_j = \{0, 1\}, \quad j = 1, \ldots, m\} \tag{A.1}$$

Unfortunately, however, there exists no efficient procedure to solve a parametric integer program, so that one needs a procedures to obtain an approximation λ_i' of $\hat{\lambda}_i$. Note that λ_i' should be no greater than λ_i in order that the resulting cut is a valid one.

Procedure 1. (LP Relaxation)

Let

$$\lambda_i' = \max \{\lambda \mid h_i'(\lambda) \leq \phi(x^*, y^*) + 1\} \tag{A.2}$$

where

$$h_i'(\lambda) = \max \{\sum_{j=1}^{n}(\delta_j + \lambda\gamma_{ij})y_j \mid \sum_{j=1}^{n} b_j y_j \leq b_0, \quad 0 \leq y_j \leq 1, \quad j = 1, \ldots, n\} + \gamma_i\lambda + \phi(\hat{x}, 0) \tag{A.3}$$

λ_i' is expected to give a good approximation of $\hat{\lambda}_i$. Moreover, $\lambda_i' \leq \hat{\lambda}_i$ since $h_i'(\lambda) \geq h_i(\lambda)$ for all $\lambda \geq 0$. What has to be solved here is a parametric linear programming problem instead of a parametric knapsack problem.

The second one is a simple search procedure which uses the convexity of h_i.

Procedure 2. (Discrete Search)

Step 0. Let $\alpha > 0$, $\beta > 1$, $\lambda = 1 + \alpha$

Step 1. Compute $g_i(\lambda)$

Step 2. If $g_i(\lambda) > \phi(x^*, y^*) + 1$, then let $\alpha = \alpha/2$, $\lambda = 1 + \alpha$ and go to Step 1. Otherwise let $\lambda = \beta\lambda$ and go to Step 3.

Step 3. Compute $g_i(\lambda)$.

Step 4. If $g_i(\lambda) \leq \phi(x^*, y^*) + 1$, then $\lambda = \beta\lambda$ and go to Step 3. Otherwise le $\lambda_i' = \lambda/\beta$ and stop.

λ_i' computed by this procedure gives an underestimate of $\hat{\lambda}_i$ for $i \in I(\hat{x})$.

Acknowledgements

I wish to thank Messrs. T. Sugiyama and M. Yoshida for their invaluable assistance in numerical experimentation.

References

[1] Balas, E. and Zemel E.: Solving Large Zero-One Knapsack Problems. MSRR 408, GSIA, Carnegie-Mellon University, (1977).

[2] Frieze, A.M.: A Bilinear Programming Formulation of the 3-dimensional Assignment Problem. *Mathematical Programming, 7* (1974), pp. 376-379.

[3] Falk, J.E.: A Linear Max-Min Problem. *Mathematical Programming, 5* (1973), pp. 169-188.

[4] Gallo, G. and Ulkücü A.: Bilinear Programming: An Exact Algorithm. *Mathematical Programming, 12* (1977), pp. 173-194.

[5] Geoffrion, A.M.: An Improved Implicit Enumeration Approach for Integer Programming. *Operations Research, 17* (1969), pp. 437-454.

[6] Glover, F.: Improved Linear Integer Programming Formulations of Nonlinear Integer Programs. *Management Science, 22* (1975), pp. 455-460.

[7] Glover, F.: Polyhedral Convexity Cuts and Negative Edge Extensions. *Zeitschrift für Operations Research, 18* (1974), pp. 181-186.

[8] Konno, H.: A Cutting Plane Algorithm for Solving Bilinear Programs. *Mathematical Programming, 11* (1976), pp. 14-27.

[9] Konno, H.: Bilinear Programming PART II: Applications of Bilinear Programming. Technical Report 71-10, Dept. of OR, Stanford University, (1971).

[10] Konno, H.: Maximizing a Convex Quadratic Function over a Hypercube. *J. of the Operation Research Society of Japan, 23* (1980), pp. 171-189.

[11] Nauss, R.M.: An Efficient Algorithm for the 0-1 Knapsack Problem. *Management Science, 23* (1976), pp. 27-31.

[12] Sherali, H.D. and C.M. Shetty: A Finitely Convergent Algorithms for Bilinear Programming Problems Using Polar Cuts and Disjunctive-Face Cuts. *Mathematical Programming, 19* (1980), pp. 14-31.

[13] Shetty, C.M. and Sherali H.D.: Rectilinear Distance Location-Allocation Problem: A Simplex Based Algorithm. *Proceedings of the International Symposium on Extreme Methods and Systems Analysis,* Springer, 1978.

[14] Vaish, H. and C.M. Shetty: The Bilinear Programming Problem. *Naval Research Logistics Quarterly, 23* (1976), pp. 303-309.

[15] Vaish, H. and Shetty, C.M.: A Cutting Plane Algorithm for the Bilinear Programming Problems. *Naval Research Logistics Quarterly, 24* (1977), pp. 83-94.

[16] Tuy, H.: Concave Programming under Linear Constraints. *Soviet Mathematics, 5* (1964), pp. 1437-1440.

Hiroshi KONNO: Institute of Information Sciences, University of Tsukuba Sakuramura, Niiharigun, Ibaraki, 305, Japan.