Journal of the Korean Institute of Industrial Engineers
Vol. 38, No. 2, pp. 74-79, June 2012.

Separation Heuristic for the Rank-1 Chvatal-Gomory Inequalities for the Binary Knapsack Problem

Kyungsik Lee†

Department of Industrial and Management Engineering, Hankuk University of Foreign Studies

이진배낭문제의 크바탈-고모리 부등식 분리문제에 대한 발견적 기법

이 경 식

한국외국어대학교 산업경영공학과

An efficient separation heuristic for the rank-1 Chvatal-Gomory cuts for the binary knapsack problem is proposed. The proposed heuristic is based on the decomposition property of the separation problem for the fixed-charge 0-1 knapsack problem characterized by Park and Lee [14]. Computational tests on the benchmark instances of the generalized assignment problem show that the proposed heuristic procedure can generate strong rank-1 C-G cuts more efficiently than the exact rank-1 C-G cut separation and the exact knapsack facet generation.

Keywords: Chvatal-Gomory cut, Knapsack problem, Separation heuristic

1. Introduction

Since the success of solving large-scale binary integer programs (BIP) by Crowder *et al.* (1983), cutting planes that can be derived from the convex hull of feasible solutions to the binary knapsack problem (BKP), so called knapsack cuts, have played a crucial role in solving BIPs. The idea of using knapsack cuts in solving BIP comes from the fact that any single constraint of BIP can be converted into a binary knapsack constraint.

The most extensively used knapsack cuts are the (lifted) cover inequalities (Balas, 1975; Balas and Zemel, 1978) that can be derived from the polyhedral structure of the knapsack polytope. The separation problem for the (lifted) cover inequality is NP-hard in general (Klabjan *et al.*, 1998), so that heuristic methods (Kaparis and Letchford, 2010) are usually used in practice. Today's leading commercial optimization softwares such as Xpress (2007) and CPLEX (2007) employ the lifted cover inequality to strengthen the bounds of the LP relaxations of the given integer programs.

Another possible way to generate knapsack cuts, though it is not widely used in practice, is to generate facet-defining inequalities of the knapsack polytope. This method is based on the observation that BKP can be solved in pseudo-polynomial time by a dynamic programming algorithm, so that we can formulate the facet generation problem for BKP as an optimization problem on the polar (Nemhauser and Wolsey, 1988) of the knapsack polytope, and the problem can be solved by a row generation method. For example, Avella *et al.* (2010) applied the facet generation method to solve the generalized assignment problem. There are two main difficulties in applying this method in practice. One is that the method requires usually too much computation time to generate violated facet-defining inequalities by a given fractional solution. The

This research was supported by Hankuk University of Foreign Studies Research Fund.

† Corresponding author : San 89, Wangsan-ri, Mohyun-myun, Yongin-shi, Kyonggi-do 449-791, Korea,
Fax : +82-30-330-4092, E-mail : globaloptima@hufs.ac.kr

Received November 25, 2011; Revision Received September 30, 2011; Accepted January 19, 2012.

other is that it is prone to round-off errors that can lead to numerical instability.

Even though the separation problem of the Chvatal-Gomory (C-G) cuts for the general integer programs is strongly NP-hard (Eisenbrand, 1999), it can be possible to generate effective C-G cuts efficiently by exploiting the structural properties of the C-G cuts for a specific problem. For example, Glover *et al*. (1997) devised a heuristic procedure for the separation of a subclass of the rank-1 C-G cuts for the multiple choice knapsack problem. They showed that the generated rank-1 C-G cuts are computationally promising. Lee and Park (2000) analyzed the separation problem for the rank-1 C-G cuts for a variable capacity 0-1 knapsack problem that includes a general integer capacity variable without upper bound. They showed that the exact separation can be done in pseudo-polynomial time. Park and Lee (2011) studied the separation problem of the rank-1 C-G cuts for the fixed-charge 0-1 knapsack problem. They showed that there exists a pseudo-polynomial time algorithm for the separation problem. They also showed the existence of a pseudo-polynomial time algorithm for the exact separation of the rank-1 C-G cuts for BKP. However, their exact separation algorithm which is based on the dynamic programming can be computationally demanding, so that an efficient separation heuristic is necessary to use the rank-1 C-G cut for BKP in practice.

In this paper, based on the results of Park and Lee (2011), we propose an efficient heuristic procedure for the separation of the rank-1 C-G cuts for BKP. We show the effectiveness of the rank-1 C-G cuts generated by proposed heuristic procedure by applying these cuts to benchmark instances of the generalized assignment problem. We also make comparisons between the strength of the heuristically separated rank-1 C-G cuts and that of other knapsack cuts including the exactly separated rank-1 C-G cuts, the lifted cover cuts, and the facet-defining inequalities of BKP.

In the next section, we present a heuristic separation procedure for the rank-1 C-G cuts for BKP. Then, we test the effectiveness of the generated cuts through computational experiment in Section 3. Finally, concluding remarks are given in Section 4.

2. Heuristic Separation Procedure

The convex hull of the feasible solutions to BKP is defined as follows :

$$KP = conv\left\{x \in B^n \mid \sum_{j \in N} a_j x_j \le b\right\},$$

where $N = \{1, \cdots, n\}$ and $a_j, j \in N$ and b are positive integers. Without loss of generality, we assume that $a_j \le b$, for all $j \in N$. Then, the rank-1 C-G inequality for KP is defined as follows :

$$\sum_{j \in N} \lfloor u_0 a_j + u_j \rfloor x_j \le \left\lfloor u_0 b + \sum_{j \in N} u_j \right\rfloor \quad (1)$$

where $u = (u_0, u_1, \cdots, u_n) \in R_+^{n+1}$, and for a given real number a, $\lfloor a \rfloor$ is the greatest integer less than or equal to a.

The convex hull of feasible solutions to the fixed-charge 0-1 knapsack problem (Park and Lee, 2011) which is closely related to KP is defined as follows :

$$FP = conv\left\{(x, y) \in B^{n+1} \mid \sum_{j \in N} a_j x_j - by \le 0\right\},$$

then KP is the projection of a face of FP, $FP \cap \{y = 1\}$, onto the x space. The rank-1 C-G inequality for FP is defined as follows :

$$\sum_{j \in N} \lfloor u_0 a_j + u_j \rfloor x_j + \lfloor -u_0 b + v \rfloor y \le \lfloor \sum_{j \in J} u_j + v \rfloor \quad (2)$$

where $u = (u_0, u_1, \cdots, u_n) \in R_+^{n+1}, v \in R_+$.

Park and Lee (2011) showed that, for a given $\overline{x} \in [0,1]^n$, if there exists a rank-1 C-G cut (1) for KP that cuts off $\overline{x}$, then there exists a rank-1 C-G cut (2) for FP that is violated by $(\overline{x}, 1)$. They also showed that the separation problem for the rank-1 C-G cut (2) for FP can be solved by a dynamic programming algorithm whose running time is $O(n^3 b^3)$. This implies that the separation problem of the rank-1 C-G cut (1) for KP can be solved by finding a rank-1 C-G cut (2) for FP, and this can be done in pseudo-polynomial time.

As mentioned above, to find a violated rank-1 C-G cut (1) with respect to $\overline{x} \in [0, 1]^n$, we have only to find a rank-1 C-G cut (2) for FP which is violated by $(\overline{x}, 1)$. Hence, the separation problem, SEP, that determine whether or not there exists a rank-1 C-G cut (2) for FP violated by $(\overline{x}, 1)$ can be formulated as the following non-differentiable optimization problem :

$$\text{(SEP)} \quad \max \sum_{j \in J} \lfloor u_0 a_j + u_j \rfloor \overline{x}_j + \lfloor -u_0 b + v \rfloor - \lfloor \sum_{j \in J} u_j + v \rfloor$$

$$\text{s. t.} \quad u \in R_+^{n+1}, \ v \in R_+,$$

where $J = \left\{j \in N \mid \overline{x}_j > 0\right\}$. Let $u_0, v, u_j, j \in J$ be an optimal solution to SEP. If the corresponding objective value is greater than or equal to 0, then the following inequality is a valid inequality for KP violated by $\overline{x}$ and it is a rank-1 C-G cut (1) for KP (Park and Lee, 2011) :

$$\sum_{j \in J} \lfloor u_0 a_j + u_j \rfloor x_j + \sum_{j \in N-J} \lfloor u_0 a_j \rfloor x_j \le \lfloor \sum_{j \in J} u_j + v \rfloor - \lfloor -u_0 b + v \rfloor .$$

Now, for a given positive integer p_0, let $p_j = \lfloor p_0 a_j / b \rfloor$, for all $j \in N$ so that $p_0 a_j / b = p_j + q_j / b$, for all $j \in N$, where q_j is an integer and $0 \leq q_j < b$. In addition, let $\mu_j = b - q_j$, for all $j \in N$, and let $\alpha_0 = 0$, $\alpha_j = \mu_j / a_j$, for all $j \in N$. Further, for given α_i, $i \in N \cup 0$, let $r_j = \lfloor (p_0 + \alpha_i) a_j / b \rfloor$ and $f_j = s_j / (b a_i) = (p_0 + \alpha_i) a_j / b - r_j$, where s_j is an integer such that $0 \leq s_j < b a_i$, for all $j \in N$. In addition, let $r_0 = \lfloor \mu_i / a_i \rfloor$ and $f_0 = s_0 / (b a_i) = \mu_i / a_i - r_0$, where s_0 is an integer such that $0 \leq s_0 < b a_i$.

One of the main results of Park and Lee (2011) is that an optimal solution to SEP can be obtained by solving SEP(p_0, α_i) defined as below, for some $1 \leq p_0 \leq b$ and α_i, $i \in N \cup 0$ such that $\alpha_i < 1$.

$$\begin{array}{lll} \text{SEP}(p_0,\ \alpha_i) & \max & \sum_{j \in J} \lfloor r_j + f_j + u_j \rfloor \overline{x}_j - \\ & & \quad \lfloor p_0 + \sum_{j \in J} u_j + r_0 + f_0 \rfloor \\ & \text{s. t.} & u_j \leq 1,\ j \in J. \end{array}$$

For a given feasible solution $\overline{u}_j$, $j \in N$, to the above SEP (p_0, α_i), the corresponding feasible solution to SEP can be recovered by setting $u_0 = (p_0 + \alpha_i)/b$, $v = \alpha_i$, and $u_j = \overline{u}_j$, for all $j \in N$.

Therefore, we can solve SEP by solving SEP(p_0, α_i) for each $1 \leq p_0 \leq b$ and α_i, $i \in N \cup 0$ such that $\alpha_i < 1$. Note that since $\overline{x}_j \leq 1$, for all $j \in J$, it can be easily seen that there exists an optimal solution to SEP(p_0, α_i) such that $u_j \in \{0, 1 - f_j\}$, for all $j \in J$. Based on these facts, we devised a heuristic procedure.

The main idea of the proposed heuristic procedure to solve SEP is to get a solution to SEP(p_0, α_i) in a greedy manner with respect to the values of $(1 - f_j)/\overline{x}_j$, $j \in J$ for each positive integer $l \leq p_0 \leq u$ and α_i, $i \in N \cup 0$ such that $\alpha_i < 1$, where l and u are prespecified positive integers. For given fractional solution $\overline{x} \in [0,1]^n$ and positive integers l and u, the proposed separation heuristic for SEP is as follows :

```
procedure SEP-HEU($\overline{x}$, $l$, $u$) :
 1. set $J = \{ j \in N \mid \overline{x}_j > 0 \}$, $Z^* \leftarrow 0$ ;
 2. for each $l \leq p_0 \leq u$ do
 3.   for each $\alpha_i$, $i \in N \cup 0$ do
 4.     set $Z(p_0) \leftarrow 0$ and
        $C = \sum_{j \in J} r_j \overline{x}_j - p_0 - r_0$ ;
 5.     sort $J$ in a decreasing order of
        $(1 - f_j)/\overline{x}_j$, $j \in J$;
 6.     let $J^* = 1, \cdots, k$ be the sorted list ;
 7.     for each $1 \leq i \leq k$ do
 8.       set $p \leftarrow \sum_{j=1}^{i} \overline{x}_j$, $w \leftarrow \sum_{j=1}^{i} (1 - f_j)$ ;
 9.       set $Z \leftarrow C + p + \lfloor w + \alpha_i \rfloor$ ;
10.       if $Z \geq Z(p_0)$ then
11.         set $Z(p_0) \leftarrow Z$ and
            $u_0 \leftarrow (p_0 + \alpha_i)/b$, $v \leftarrow \alpha_i$ ;
12.         set $u_j \leftarrow (1 - f_j)$, $j \leq i$ and
            $u_j \leftarrow 0$, $j \geq i + 1$ ;
13.       end if
14.     end for
15.     if $Z(p_0) \geq Z^*$ then
16.       set $Z^* \leftarrow Z(p_0)$, $u_0 \leftarrow \overline{u_0}$, $v \leftarrow \overline{v}$,
          $u_j \leftarrow \overline{u_j}$, $j \in J$;
17.     end if
18.   end for
19. end for
end procedure
```

In the above procedure SEP-HEU($\overline{x}$, l, u), to make a sorted list J^* in the line 5, for a given pair of p_0 and α_i, is the most time consuming step, which can be done in $O(n \log n)$. Therefore, overall complexity of SEP-HEU($\overline{x}$, l, u) is $O(mn^2 \log n)$, where $m = u - l + 1$.

In the next section, we show that the proposed separation heuristic is effective and much less computationally demanding than the dynamic programming based exact separation algorithm (Park and Lee, 2011).

3. Computational Results

To test the performance of the rank-1 C-G cut for BKP and the separation heuristic proposed in the previous section, we choose the benchmark problem set in the OR-Library (Beasley, 1990). In particular, we perform the tests on the problem instances of the generalized assignment problem (GAP) which is defined as follows :

$$\begin{array}{lll} \text{(GAP)} & \min & \sum_{i \in M} \sum_{j \in N} c_{ij} x_{ij} \\ & \text{s. t.} & \sum_{i \in M} x_{ij} = 1, \text{ for all } j \in N, \\ & & \sum_{j \in N} a_{ij} x_{ij} \leq b_i, \text{ for all } i \in M, \\ & & x_{ij} \text{ binary, for all } i \in M,\ j \in N. \end{array}$$

There exist $|M|$ binary knapsack constraints in GAP, and we generate the rank-1 C-G cuts for each of these constraints. We choose GAP as our benchmark test set since it is a well-known difficult problem which has been studied extensively, for example see (Avella *et al.*, 2010). In the computational test, we generate cuts from each knapsack constraint of GAP for a given fractional solution. To generate the rank-1 C-G cut by using the proposed heuristic separation procedure SEP_HEU($\overline{x}, l, u$), we set $l = 1$ and $u = \min(|N|,\ b_i - 1)$, for each $i \in M$.

3.1 The Performance of the Proposed Separation Heuristic Procedure

First, we tested the performance of the proposed separation heuristic on the GAP instances in OR-Library (Beasley, 1990). The GAP instances consist of five classes (A, B, C, D, E). Instances of the classes A and B are known to be easily solved (Beasley, 1990) and so, we do not consider them. We choose instances in the classes C, D, and E with sizes from 5-by-100 to 40-by-400, where the first number is the number of knapsack constraints and the second is that of binary variables in each knapsack constraint. To compare with other knapsack cuts, the final values of LP relaxations at the root node of the branch-and-bound tree are compared with those obtained by the exact knapsack facet generation procedure and commercial optimization software (Xpress, 2007; CPLEX, 2007). Note that the two optimization softwares automatically add lifted cover inequalities. The exact knapsack facet generation results can be found in (Avella *et al.*, 2010), where the row generation procedure was used to solve the problem. To implement heuristic separation procedure, we used Mosel language and LP solver provided by Xpress (2007).

<Table 1> shows the results. In the table, the heading Closed Gap' refers to percentage of the gap closed, formally it is defined as :

$$\text{Closed Gap} = \frac{LB - LB_{LP}}{UB - LB_{LP}} \times 100(\%),$$

where LB_{LP} is the value of initial *LP* relaxation, *UB* is the optimal *IP* value (for those instances where the optimal values are unknown, it refers to the best known value), and *LB* is the value of *LP* relaxation strengthened by the corresponding knapsack cuts (KF : exact knapsack facet (Avella *et al.*, 2010), HCG : heuristically separated rank-1 C-G cuts, XLC : lifted cover cuts by Xpress (2007), CLC : lifted cover cuts by CPLEX (2007)).

In the table, we can see that the performance of the proposed heuristic as measured by the closed gap outperforms those of two commercial MIP solvers. Also, it is comparable with that using the exact knapsack facet generation procedure. It shows that about 82% of the closed gap by the exact knapsack facet generation can be recovered by the heuristically generated rank-1 C-G cuts. XLC and CLC closed about 56% and 40% of the closed gap by KF, respectively. The results show that the simple heuristic proposed in the previous section can be quite effective, and outperform the performance of the lifted cover cuts.

Table 1. Performance of the Heuristic Separation Procedure

Instance	Closed Gap (%)			
	KF	HCG	XLC	CLC
c05100	85.7	77.6	47.5	36.4
c10100	85.7	84.8	68.0	61.0
c20100	95.8	89.1	88.8	81.7
c05200	80.0	60.8	39.6	23.0
c10200	80.0	75.3	55.4	30.0
c20200	100.0	89.7	68.6	60.1
c10400	80.0	68.5	45.9	29.7
c20400	85.7	70.7	46.1	30.5
c40400	100.0	91.6	78.1	67.4
d05100	57.1	33.4	16.5	9.5
d10100	78.3	47.2	20.3	14.4
d20100	81.1	53.9	31.9	22.5
d05200	80.0	48.3	17.4	10.7
d10200	63.6	39.7	16.7	5.1
d20200	46.2	36.1	17.5	13.9
d10400	23.1	13.1	7.0	1.8
d20400	25.0	19.0	9.0	3.8
d40400	1.4	1.3	0.6	0.4
e05100	82.1	63.1	33.9	30.0
e10100	75.8	63.4	42.0	35.3
e20100	94.7	86.3	65.6	58.3
e05200	62.5	50.0	30.4	16.4
e10200	69.2	55.7	48.31	36.3
e20200	91.3	84.4	61.2	38.1
e10400	83.3	70.6	52.9	22.5
e20400	93.3	84.2	48.7	21.6
e40400	89.2	81.1	54.3	36.9
Avg	73.7	60.7	41.2	29.5
Min	1.4	1.3	0.6	0.4
Max	100.0	98.9	88.8	81.7

3.2 Comparison with the Exact Rank-1 C-G Cut and the Knapsack Facet

To make comparisons between the strength of the rank-1 C-G cuts separated by the proposed heuristic procedure and those of the exact rank-1 C-G cuts and the knapsack facets, we implemented the exact rank-1 C-G cut separation procedure (Park and Lee, 2011) and the exact knapsack facet generation procedure (Avella *et al.*, 2010) by using the Mosel language (Xpress, 2007). We tested the procedures on small instances of GAP in OR-Library (Beasley, 1990) since the exact rank-1 C-G separation procedure and the exact knapsack facet generation procedure require relatively much computation time. The results are summarized in <Table 2>. The heading 'ECG' refers to the exact rank-1 C-G separation procedure, and the meaning of

'Closed Gap' is the same as that defined in Section 4.1. The heading 'Time' refers to the CPU seconds taken by each separation procedure.

Table 2. Comparisons with the Exact Rank-1 C-G Cut and the Exact Knapsack Facet

Instance	HCG		ECG		KF	
	Closed Gap	Time	Closed Gap	Time	Closed Gap	Time
c1030-1	83.53	1.21	89.02	12.80	89.02	12.58
c1030-2	92.89	1.28	96.57	8.33	96.57	17.28
c1030-3	82.91	1.40	84.56	7.78	89.71	19.64
c1030-4	81.40	0.62	81.40	4.23	81.40	8.08
c1030-5	84.69	1.28	87.47	10.56	87.47	18.25
c1040-1	99.79	1.89	100.00	8.27	100.00	22.17
c1040-2	87.77	2.03	87.77	15.27	90.21	91.50
c1040-3	93.76	1.90	97.00	13.56	98.07	73.00
c1040-4	87.67	1.89	100.00	15.59	100.00	99.03
c1040-5	78.48	2.57	86.04	24.80	86.16	90.41
c1050-1	89.55	2.04	93.03	23.14	93.03	149.53
c1050-2	94.72	2.84	95.06	30.94	97.10	171.70
c1050-3	93.06	2.42	93.06	12.53	93.06	146.74
c1050-4	84.45	3.54	85.37	35.48	88.48	237.83
c1050-5	74.86	2.84	75.23	25.88	75.77	195.70
c1060-1	100.00	4.56	100.00	45.75	100.00	601.00
c1060-2	80.37	4.45	81.40	108.68	82.02	437.03
c1060-3	78.33	4.00	80.42	54.64	86.16	670.06
c1060-4	78.76	2.84	80.72	55.92	81.05	427.42
c1060-5	91.54	4.12	91.54	21.09	91.54	607.04
Average	86.93	2.49	89.28	26.76	90.34	204.80

From <Table 2>, we can see that the performance of heuristic separation procedure is almost the same as that of the exact rank-1 cut generation procedure. On average, the gap closed by the exact rank-1 C-G cut is 89.28% (with 26.76 seconds of the average computation time), whereas that by the heuristic is around 86.93% (with 2.49 seconds of the average computation time). For reference, the average gap closed by the exact knapsack facets is 90.34% and the procedure took 204.80 seconds on average. Even though the computation time can be significantly enhanced if we implement each separation procedure by using other programming languages such as C and C++ instead of the relatively slow Mosel language, the results clearly show that the proposed heuristic separation procedure is more efficient than the other separation procedures and the generated rank-1 C-G cuts are effective.

<Table 3> shows the number of cut generation rounds (the heading #R) taken by each separation procedure to reach the closed gap reported in <Table 2> and the number of generated cuts (the heading #C) by each procedure.

We can see that the numbers of cuts generated by the heuristic (average 80.25) and the exact rank-1 separation procedure (average 79.95) are almost the same, whereas the number of cuts generated by the exact knapsack facet generation procedure is relatively large (average 217.16).

We can also see that the number of cut generation rounds taken by the exact knapsack facet generation procedure (average 36.70) is more than two times larger than those of the other two separation procedures. This result may indicate the rate of gap reduction of the exact facet generation is slower than those of the other two separation procedures.

Table 3. The Number of Cut Generation Rounds and the Number of Added Cuts

Instance	HCG		ECG		KF	
	#R	#C	#R	#C	#R	#C
c1030-1	13	71	18	80	23	163
c1030-2	13	75	12	68	31	200
c1030-3	15	66	12	57	34	152
c1030-4	7	49	7	41	15	90
c1030-5	13	71	15	79	31	224
c1040-1	16	87	9	62	11	92
c1040-2	15	81	14	85	39	258
c1040-3	14	76	13	85	34	225
c1040-4	15	75	14	85	43	248
c1040-5	18	110	17	107	42	307
c1050-1	12	78	16	84	29	216
c1050-2	17	91	17	85	33	154
c1050-3	18	50	14	46	30	79
c1050-4	20	84	17	86	31	220
c1050-5	17	84	15	78	36	249
c1060-1	20	115	18	103	57	371
c1060-2	21	89	19	99	36	235
c1060-3	18	99	15	96	84	448
c1060-4	13	62	16	85	32	195
c1060-5	18	92	14	88	63	322
Average	15.65	80.25	14.60	79.95	36.70	222.40

To further investigate the rate of gap reduction of each separation procedure, we compared the closed gap in each round of cut generation. The test was done on 5 instances of GAP (c1060-1, c1060-2, c1060-3, c1060-4, and c1060-5). The result for each instance is almost similar to the others, so the result for the instance c1060-1 is reported in <Table 4>. <Table 4> shows the closed gap by each separation procedure in each cut generation rounds (the heading 'ROUND') for 15 cut generation rounds.

For the instance c1060-1, the numbers of rank-1 C-G cuts generated by the heuristic procedure and the exact separation procedure in 15 rounds was 100 and 92, respectively, whereas that of the exact knapsack facet generation procedure was 144.

As we can see from <Table 4>, the gap reduction of rank-1 C-G inequalities (both heuristic and exact) is much faster than that of the exact knapsack facet generation.

Table 4. The Rate of Gap Reduction for c1060-1

ROUND	Closed Gap(%)		
	HCG	ECG	KF
1	20.52	20.52	21.50
2	40.72	39.74	28.01
3	55.70	54.07	34.85
4	75.57	71.99	41.04
5	78.50	76.55	45.93
6	80.46	81.11	49.84
7	83.39	83.71	54.40
8	84.69	85.99	60.26
9	86.32	86.64	63.19
10	88.27	89.25	69.06
11	89.90	96.42	72.96
12	91.21	100.00	77.20
13	95.44	100.00	79.48
14	100.00	100.00	84.04
15	100.00	100.00	85.99

4. Concluding Remarks

This paper proposed an efficient heuristic procedure for the separation of the rank-1 C-G cuts for BKP that can be used as a cut generation method for general 0-1 integer programs, in which every constraint can be transformed into a 0-1 knapsack constraint. The computational results show that the proposed heuristic can be readily adopted in commercial MIP solver to strengthen the LP relaxations. Also, the method can be used as a preprocessing procedure to strengthen the quality of MIP formulations.

The computational test in this paper was done on GAP instances, so more extensive tests on other benchmark instances such as those in MIPLIB 2003 (Achterberg *et al*., 2003) would be needed to further investigate the performance of the separation procedure. Also, more sophisticated implementation of the procedure would be possible to speed up the computation time.

References

Achterberg, T., Koch, T., and Martin, A. (2003), MIPLIB 2003, *Operations Research Letters*, **34**, 361-372.

Avella, P., Boccia, M., and Vailyev, I. (2010), A computational study of exact knapsack separation for the generalized assignment problem, *Computational Optimization and Applications*, **45**, 543-555.

Balas, E. (1975), Facets of the knapsack polytope, *Mathematical Programming*, **8**, 146-164.

Balas, E. and Zemel, E. (1978), Facets of the knapsack polytope from minimal covers, *SIAM Journal on Applied Mathematics*, **34**, 119-148.

Beasley, J. E. (1990), OR-Library : Distributing test problems by electronic mail, *Journal of the Operational Research Society*, **41**, 1069-1072.

CPLEX 9.1, http://www.ibm.com, 2007.

Crowder, H., Johnson, E., and Padberg, M. (1983), Solving large-scale 0-1 linear programming problems, *Operations Research*, **31**, 803-834.

Eisenbrand, F. (1999), On the Membership Problem for the Elementary Closure of a Polyhedron, *Combinatorica*, **19**, 297-300.

Glover, F., Sherali, H. D., and Lee, Y. (1997), Generating Cuts from Surrogate Constraint Analysis for Zero-One and Multiple Choice Programming, *Computational Optimization and Application*, **8**, 151-172.

Kaparis, K. and Letchford, A. N. (2010), Separation algorithms for 0-1 knapsack polytopes, *Mathematical Programming*, **124**, 69-91.

Klabjan, D., Nemhauser, G. L., and Tovey, C. (1998), The complexity of cover inequality separation, *Operations Research Letters*, **23**, 35-40.

Lee, K. and Park, S. (2000), A Cut Generation Method for the (0, 1)-Knapsack Problem with a Variable Capacity, *Journal of the Korean OR/MS Society*, **25**(3), 1-15.

Nemhauser, G. L. and Wolsey, L. A. (1988), Integer and Combinatorial Optimization, Wiley.

Park, K. and Lee, K. (2011), On the separation of the rank-1 Chvatal-Gomory Inequalities for the fixed-charge 0-1 knapsack problem, *Journal of the Korean OR/MS Society*, **36**(2), 43-50.

Xpress Optimizer 17.10.04, http://www.fico.com, 2007.