Local Cuts for 0–1 Multidimensional Knapsack Problems

Hanyu Gu

Abstract This paper investigates the local cuts approach for the multidimensional knapsack problem (MKP) which has more than one knapsack constraints. The implementation of the local cuts-based cutting plane algorithm is an extension of the exact knapsack separation scheme of Vasilyev et al. (J Glob Optim 1–24, [13]). Comparisons are made with the global lifted cover inequalities (GLCI) proposed in our recent paper in Computers & Operations Research (Gu, Comput Oper Res 71:82–89, [7]). Preliminary results show that the local cuts approach may be powerful for the MKP.

Keywords Multidimensional Knapsack Problem · Cutting plane · Local cuts

1 Introduction

The multidimensional knapsack problem (MKP) is a well-known strongly NP-hard combinatorial optimization problem which has been applied to various resource allocation-related practical problems [4]. Given m knapsacks with capacities b_i, $i = 1, \dots, m$, and n items contributing profits c_j, $j = 1, \dots, n$, the MKP maximizes the total profits of selected items under the knapsack capacity limitations. In MKP, an item j requires simultaneous resource consumption of $a_{i,j}$ units in the ith knapsack ($i = 1, \dots, m$, $j = 1, \dots, n$). The MKP can be formulated as an integer linear programming (ILP):

$$\text{(MKP)} \qquad z^* = \max\{c^T x : Ax \le b, x \in \{0,1\}^n\} \tag{1}$$

where $c = [c_1, c_2, \dots, c_n]^T$ is the profit vector; $x = [x_1, x_2, \dots, x_n]^T$ is a vector of 0-1 decision variables; x_i is equal to 1 if item i is selected, otherwise 0; $A = [a_{i,j}]$, $i = 1, 2, \dots, m$, $j = 1, 2, \dots, n$ is the resource consumption matrix; $b = [b_1, b_2, \dots, b_m]^T$

H. Gu (✉)
School of Mathematical and Physical Sciences, University of Technology Sydney,
15 Broadway, Ultimo, NSW 2007, Australia
e-mail: Hanyu.Gu@uts.edu.au

R. Sarker et al. (eds.), *Data and Decision Sciences in Action*,
Lecture Notes in Management and Industrial Engineering,
DOI 10.1007/978-3-319-55914-8_6

is the vector of knapsack capacities. For MKP, all parameters are assumed to be nonnegative integers.

A large amount of research efforts has been committed to MKP which is closely related to the classical knapsack problem (KP) [12]. In spite of the huge theoretical and computational progress in combinatorial optimization, MKP remains a challenging problem [5, 14]. It still takes many hours on a modern computer to obtain the best-known solutions for some of the well-known benchmark problems.

In this paper, we focus on the cutting plane approach which has found tremendous success in the commercial and open-source mixed integer programming (MIP) solvers. For knapsack problem, the cover-based valid inequalities are among the best-studied classes and have been shown useful, especially for solving difficult binary integer programming problems including the MKP [8, 11]. Recently, exact knapsack separation algorithms (EKSA) were proposed in [6, 11]. Although EKSAs can produce stronger valid inequalities, they are more or less useless in the branch and cut paradigm due to much longer computation time. An efficient EKSA was implemented in [13] and was found useful for solving large instances of some classical MIP problems.

Recently, the global lifted cover inequalities (GLCI) [10] and its variants were proposed for MKP to take into account multiple knapsack constraints simultaneously when lifting the coefficients of a valid inequality. The face defined by GLCI may not be of higher dimensions than lifted cover inequalities (LCI) for MKP; however, computational results demonstrated that GLCI can be more effective in tightening the integrality gap, especially when the number of knapsack constraints increases. Furthermore, the computationally stronger GLCI can lead to improved performance of core-based heuristics [7]. This motivates the research in this paper to combine the power of exact separation and global lifting, which the authors' believe belongs to the local cuts paradigm which is very different from the traditional template paradigm [2].

This paper is organized as follows. We first describe the cutting plane algorithm based on the local cuts approach in Sect. 2. The details of the exact separation for the reduced problem is given in Sect. 3. The global lifting procedure is described in Sect. 4. Computational results are presented in Sect. 5. The conclusion is given in Sect. 6.

2 Cutting Plane Algorithm Based on Local Cuts

The cutting plane approach [15] can be used to strengthen the linear programming (LP) relaxation of MKP which is defined as

$$\text{(MKP-LP)} \qquad \bar{z} = \max\{c^T x : Ax \le b, x \in [0,1]^n\} \tag{2}$$

MKP-LP is solved at each iteration of the cutting plane approach with additional constraints corresponding to the cuts generated from previous iterations. Assume that an optimal continuous solution x^k is obtained at the kth iteration, an attempt will be made to generate a cut separating x^k from the convex hull of the MKP polytope. If cuts are successfully generated, they are added to MKP-LP and the process is repeated until some termination criteria are met. Since the separation algorithm can be time-consuming in practice, this process has to be terminated earlier without the exact description of the convex hull of the MKP polytope. In the local cuts approach, a reduced problem of MKP (RMKP) is created at each iteration which can be efficiently separated due to much lower dimension. A valid cut for the original problem can then be induced. The cutting plane algorithm based on local cuts (CPALC) is described as follows:

Algorithm 1: Cutting Plane Algorithm Based on Local Cuts (CPALC)

```
Input: MKP
Output: a set of valid inequalities for MKP
$k = 1$;
repeat
    solve the cut strengthened MKP-LP with optimal solution $x^k$;
    if $x^k$ is an integer solution then break;
    generate the separation problem for the reduced MKP (RMKP) problem;
    compute cuts for RMKP;
    induce valid cuts for MKP to separate $x^k$;
    if no cut then break;
    add cuts to MKP-LP;
until enough cuts or time limit reached;
return all the generated cuts;
```

The creation of the separation problem for RMKP is described in the next section. A cut for the original MKP can always be induced from a cut for RMKP through the global lifting procedure discussed in Sect. 4.

3 Exact Separation Algorithm for RMKP

Given x^k, the LP relation solution of MKP in CPALC, let $V^k = \{i|x_i^k = 1\}$, and $W^k = \{i|x_i^k = 0\}$. Denote $R^k = N \setminus (V^k \cup W^k) = \{r_1, r_2, \dots, r_p\}$, the RMKP is defined as

$$\begin{aligned} \text{RMKP} \quad \max \quad & z = \sum_{1\le j\le p} c_{r_j} y_j \\ \text{s. t.} \quad & \sum_{1\le j\le p} a_{ir_j} y_j \le \tilde{b}_i^k, \quad i = 1, \dots, m \\ & y_j \in \{0, 1\}, \quad 1 \le j \le p \end{aligned}$$

with $\tilde{b}_i^k = b_i - \sum_{j\in V^k} a_{ij}$, $i = 1, \dots, m$.

The separation problem for RMKP is to separate y^k defined as $y^k_j = x^k_{r_j}, 1 \leq j \leq p$. The RMKP polytope has much lower dimension than that of MKP, which may be amenable to the exact separation approach. The exact separation formulation can take many different forms. Our implementation of the exact separation for RMKP is an adaptation of the source code of [13] for single knapsack problem and is based on the formulation below [13]:

$$\begin{aligned} \text{RMKP-S} \quad \max_{\pi} \quad & z = \pi^T y^k \\ \text{s. t.} \quad & h^T \pi \leq 1, \quad \forall h \in P \\ & \pi \in \Pi \end{aligned}$$

where $P \neq \emptyset$ is the feasible set of RMKP, and $\Pi \subset \mathscr{R}^p$ is a sufficiently large convex compact set containing the origin in its interior. Denote the optimal solution to RMKP-S by $\bar{\pi}^k$, then $y^T \bar{\pi}^k \leq 1$ is a valid cut separating y^k from the RMKP polytope if $y^{kT} \bar{\pi}^k > 1$. The nice property of RMKP-S is that the cut $y^T \bar{\pi}^k \leq 1$ also defines a facet of RMKP.

RMKP-S can be solved by row generation. At the lth iteration, π^l is solved as an optimal solution to

$$\begin{aligned} \text{RMKP-S}(l) \quad \max_{\pi} \quad & z = \pi^T y^k \\ \text{s. t.} \quad & h^T \pi \leq 1, \quad \forall h \in P^l \subset P \\ & \pi \in \Pi \end{aligned}$$

If $y^{kT} \pi^l \leq 1$, y^k is an interior point of the RMKP polytope since $y^{kT} \pi^l \geq y^T \bar{\pi}^k$. Otherwise, a new row may be obtained by solving

$$\begin{aligned} \text{Row-Gen} \quad \max_{h} \quad & z = h^T \pi^l \\ \text{s. t.} \quad & h \in P \end{aligned}$$

which has optimal objective value z^l and optimal solution h^l.

If $z^l \leq 1$, then $\bar{\pi}^k = \pi^l$ and $y^T \pi^l \leq 1$ is a cut for RMKP; otherwise, h^l can be added to P^l and a new row is generated for RMKP-S(l).

To avoid numerical errors, all coefficients in the cut need to be converted into integer. In [13], $\bar{\pi}^k$ is multiplied by a proper integer to make it an integer vector $\tilde{\pi}^k$ (within a tolerance of 10^{-5}). The separation procedure fails if no such integer multiplier can be found within the range $[1, \ldots, 10^4]$. Then, the right-hand side can be calculated as

$$\pi^k_0 = \max\{h^T \tilde{\pi}^k \, : \, h \in P\} \tag{3}$$

$y^T \tilde{\pi}^k \leq \pi^k_0$ is a valid cut for RMKP if it is violated by y^k.

If y_j can be fixed to 0 in RMKP, i.e., $\exists i \in [1, \dots, m]$, $a_{i,r_j} > \tilde{b}_i^k$, we can generate a cut for RMKP in the form $y_j \leq 0$ instead of solving RMKP-S which will set $\bar{\pi}_j^k$ to the largest value of π_j in Π.

The exact separation algorithm for RMKP (ESAR) can be described as follows:

Algorithm 2: Exact Separation Algorithm for RMKP (ESAR)

```
Input: RMKP, $y^k$
Output: a set of cuts separating $y^k$ from RMKP polytope
compute $F = \{j \in [1, \dots, p] : a_{i,r_j} > \tilde{b}_i^k, \exists i \in [1, \dots, m]\}$;
if $F \neq \emptyset$ then
    generate cuts $y_j \leq 0, \forall j \in F$;
    return
$l = 1, P^1 = \{e_1, \dots, e_p\}$ where $e_i$ is the $i$th unit vector;
Loop
    solve RMKP-S($l$) with the optimal solution $\pi^l$;
    if $y^{kT}\pi^l \leq 1$ then return //$y^k$ is interior point
    solve Row-Gen with optimal value $z^l$ and optimal solution $h^l$;
    if $z^l \leq 1$ then
        $\bar{\pi}^k = \pi^l$; //$y^T\pi^l \leq 1$ is a cut for RMKP
        break;
    $P^{l+1} = P^l \cup \{h^l\}, l = l + 1$;
compute the integral inequality $y^T\tilde{\pi}^k \leq \pi_0^k$ according to (3);
generate a cut $y^T\tilde{\pi}^k \leq \pi_0^k$ if it is violated by $y^k$;
return
```

4 Global Lifting Algorithm

The valid inequality for RMKP generated by ESAR can be sequentially lifted to become valid for MKP. We apply the same idea as for GLCI in [7] by considering all the constraints in MKP when lifting a variable.

Let K be the feasible set of MKP, and $S(V, W) = K \cap \{x \in \{0, 1\}^n | x_j = 0, \forall j \in W, x_j = 1, \forall j \in V\}$. A variable in V can be down-lifted according to the following proposition:

Proposition 1 *Given $\sum_{j \in N \setminus (V \cup W)} \pi_j x_j \leq \pi_0$ is valid for $S(V, W)$. For $i \in V$, if $S(V \setminus \{i\}, W) \neq \emptyset$, then $\gamma_i x_i + \sum_{j \in N \setminus (V \cup W)} \pi_j x_j \leq \pi_0 + \gamma_i$ is valid for $S(V \setminus \{i\}, W)$ for any $\gamma_i \geq \zeta - \pi_0$, where*

$$\zeta = \max\{\sum_{j \in N \setminus (V \cup W)} \pi_j x_j | x_i = 0, x \in S(V \setminus \{i\}, W)\} \tag{4}$$

A variable in W can be up-lifted according to the following proposition:

Proposition 2 *Given* $\sum_{j\in N\setminus(V\cup W)} \pi_j x_j \le \pi_0$ *is valid for* $S(V, W)$*. For* $i \in W$*, if* $S(V, W \setminus \{i\}) \ne \emptyset$*, then* $\alpha_i x_i + \sum_{j\in N\setminus(V\cup W)} \pi_j x_j \le \pi_0$ *is valid for* $S(V, W \setminus \{i\})$ *for any* $\alpha_i \le \pi_0 - \zeta$*, where*

$$\zeta = \max\{ \sum_{j\in N\setminus(V\cup W)} \pi_j x_j | x_i = 1,\, x \in S(V, W \setminus \{i\})\} \tag{5}$$

For a cut generated by ESAR in the form of $y^T\tilde{\pi}^k \le \pi_0^k$, we can have $\sum_{1\le j\le p} x_{r_j}^T \hat{\pi}_{x_j}^k \le \pi_0^k$ with $\hat{\pi}_{x_j} = \tilde{\pi}_j^k$, which is valid for $S(V^k, W^k)$. Then, the variables in V^k are down-lifted sequentially in the non-decreasing order of the reduced costs obtained when solving for x^k at step 3 of Algorithm 2. Finally, the variables in W^k are up-lifted sequentially in the non-increasing order of the reduced costs. To reduce computation time on lifting, we calculate the down-lifting coefficient according to $\gamma_i = \zeta_{LP} - \pi_0$, and the up-lifting coefficient according to $\alpha_i = \pi_0 - \zeta_{LP}$, where ζ_{LP} is the linear relaxation upper bound of ζ in (4) and (5).

Denote the valid inequality after lifting by $x^T\hat{\pi}^k \le \hat{\pi}_0^k$, we have

$$x^{kT}\hat{\pi}^k - \hat{\pi}_0^k = y^{kT}\tilde{\pi}^k - \pi_0^k \tag{6}$$

Therefore, the cut for RMKP is also a cut for MKP after the lifting.

5 Computational Results

We first compare the exact knapsack separation algorithm (EKSA) of [13] with our implementations of GLCI and its variants [7]. Then, EKSA is compared with the local cuts (LC)-based separation algorithm CPALC implemented in this paper. All experiments are carried out on the comprehensive Cho test set in [1] and the widely used Chu and Beasley MKP test set in [3]. The algorithms presented in this paper were implemented in C++ and compiled with gcc 4.4. All tests were run on a computer with 2.3 GHz AMD CPU and Red Hat 4.4 operating system. We use CPLEX 12.5 to solve the LP and IP problems in CPALC with a time limit of 60 s and a limit on the maximum number of threads to 8. The number of cuts in CPALC is limited to 100 as for GLCI, but there is no limit for the number of cuts in EKSA which typically takes no more than a few seconds. In RMKP-S(l), Π is set to be $[-10^6,\ 10^6]^p$.

The Cho test set contains classes of randomly generated instances for each combination of $n \in \{50, 100, 250\}$ items and $m \in \{5, 10, 30\}$ constraints. Each class contains 30 instances. The Cho test set represents a much larger range of correlation values than the Chu and Beasley test set under the MKP structure and was created according to a well-designed comprehensive problem generation scheme [9]. The results of GLCIs and EKSA for the Cho test set are reported in Table 1. The columns in Table 1 are explained as follows: The first column *n.m* represents the class of test cases where n is the number items and m is the number of knapsack constraints; the

Table 1 Comparison of GLCI and EKSA on the Cho test set [1]

$n.m$	z^*	$\bar{z}$	Δ_D	GSA-LP			GSA-IP			EKSA		
				I_C	N_c	T_v	I_C	N_c	T_v	I_C	N_c	T_v
50.5	1481.6	1494.6	0.91	51.9	13.0	–	49.5	12.9	0.2	78.6	43.7	0.1
50.10	1199.2	1214.9	1.32	50.2	19.9	0.1	53.4	22.6	1.3	66.2	53.7	0.1
50.25	968.7	984.6	1.63	46.9	29.8	0.2	49.6	37.9	7.4	57.2	63.8	0.4
100.5	2824.7	2830.8	0.22	41.8	9.2	–	44.2	10.0	0.5	83.3	62.6	0.2
100.10	2548.2	2558.8	0.42	33.3	15.8	–	33.5	18.6	1.9	59.9	76.1	0.4
100.25	1959.5	1972.3	0.64	34.1	23.6	0.2	37.4	30.0	8.2	50.7	82.5	0.6
250.5	7036.3	7040.2	0.05	41.2	9.1	0.1	42.3	9.3	0.7	74.8	90.6	0.7
250.10	6097.8	6103.2	0.09	26.2	12.5	0.2	28.2	13.2	2.2	58.7	107.6	0.9
250.25	4874.6	4882.7	0.16	22.0	17.7	0.6	23.9	22.2	17.5	38.6	113.1	1.3

second column reports the best-known objective value z^* found in the literature; the next column presents the objective value of the LP relaxation $\bar{z}$ without additional cuts; the fourth column gives the relative integrality gap $\Delta_D = (\bar{z} - z^*) \times 100/\bar{z}\%$; the next column reports the closed integrality gap in percentage; GSA-LP and GSA-IP are the different implementations of GLCI in [7]; and the next two columns present the total number of cuts generated by the corresponding separation scheme and the total CPU time spent T_v in seconds. The value of T_v is replaced with "–" if it is smaller than 0.1 s. For each class of test cases, the arithmetic mean of the 30 instances of the same class is reported. It can be seen that EKSA significantly outperforms GLCIs in terms of the percentage of integrality gap closed. The computation time of EKSA is only slightly higher than that of GSA-LP. The superiority of EKSA may be due to the weak interactions of the knapsack constraints in the instances.

The results of EKSA and CPALC for the Cho test set are reported in Table 2. The time for the exact separation in Sect. 3 and the time for the global lifting in Sect. 4 are

Table 2 Comparison of EKSA and CPALC on the Cho test set [1]

$n.m$	z^*	$\bar{z}$	Δ_D	EKSA			CPALC				
				I_C	N_c	T_v	I_C	N_c	T_v	T_c	T_l
50.5	1481.6	1494.6	0.91	78.6	43.7	0.1	100.0	48.2	11.7	11.5	0.1
50.10	1199.2	1214.9	1.32	66.2	53.7	0.1	100.0	55.6	18.3	18.2	0.1
50.25	968.7	984.6	1.63	57.2	63.8	0.4	98.7	70.9	69.6	69.3	0.3
100.5	2824.7	2830.8	0.22	83.3	62.6	0.2	100.0	64.1	13.4	12.6	0.8
100.10	2548.2	2558.8	0.42	59.9	76.1	0.4	92.8	80.8	61.0	60.1	0.9
100.25	1959.5	1972.3	0.64	50.7	82.5	0.6	86.4	88.7	125.9	124.5	1.5
250.5	7036.3	7040.2	0.05	74.8	90.6	0.7	92.8	82.4	41.8	38.1	3.7
250.10	6097.8	6103.2	0.09	58.7	107.6	0.9	84.4	95.0	64.3	60.4	3.9
250.25	4874.6	4882.7	0.16	38.6	113.1	1.3	64.6	101.0	176.7	170.0	6.7

Table 3 Comparison of GLCI, EKSA, and CPALC on 100-item instances from Chu and Beasley test set [3]

$m.\alpha$	GSA-LP			GSA-IP			EKSA			CPALC		
	I_C	N_c	T_v	I_C	N_c	T_v	I_C	N_c	T_v	I_C	N_c	T_v
5. 25	6.99	15.8	–	6.80	16.3	0.4	19.18	119.5	1.6	41.85	101	239.5
5. 50	6.11	13	–	6.38	14.4	0.2	22.65	122.6	2.7	44.54	101	256.8
5. 75	6.50	12.7	–	7.46	12.7	0.2	23.52	121.4	3.7	51.33	101	241.1
10. 25	6.49	19	–	8.78	37.3	9.1	5.9	100.5	1.9	24.88	76.6	754.4
10. 50	6.96	21.8	–	10.28	30	4.5	7.83	120.7	3.7	28.06	80	1070.1
10. 75	4.18	13.2	–	5.07	14.8	2	7.62	97.5	3.5	32.15	79.2	908.4
30. 25	2.74	15.5	1.1	6.49	73.1	269.9	0.22	12.3	1.6	6.17	10.5	645.4
30. 50	5.30	33.5	2.3	7.79	90.5	233.1	0.55	38.4	2.8	5.89	8.4	323.7
30. 75	4.84	28.4	2.5	11.26	86.7	230.5	0.63	37	2.9	11.09	15.8	1286.8

reported as T_e and T_l in seconds. It can be seen that CPALC significantly outperforms EKSA in terms of closed integrality gap. The computation time of CPALC is much higher than that of EKSA. The global lifting is very efficient in terms of computation time. EKSA failed to close integrity gap on many instances even without the limit on the number of cuts. CPALC can always separate the LP solution which is consistent with the theory; however, the convergence can be slow on large instances with more constraints.

The Chu and Beasley test set contains a total of 270 instances randomly generated for 27 classes which are combinations of the number of items $n \in \{100, 250, 500\}$, the number of knapsack constraints $m \in \{5, 10, 25\}$, and knapsack capacity tightness ratios $\alpha \in \{0.25, 0.5, 0.75\}$. The results for the Chu and Beasley test set are reported in Table 3 for test cases with 100 items. For each class of test cases, the arithmetic mean of the 10 instances of the same class is reported. For cases with five constraints, EKSA closed much higher integrity gaps than GLCIs with marginally longer computation time. However, GLCIs perform significantly better than EKSA when $m = 30$. The reason could be that the intersection of single knapsack polytopes is not a good approximation of the MKP polytope as the number of constraints increases. The exact separation procedure of CPALC failed on many cases when $m = 10$ and $m = 30$ due to the difficulty to make integral the cutting plane obtained from RMKP-S(l). However, CPALC achieved similar amount of closed integrity gaps with many fewer cuts when compared with GSA-IP, which demonstrated the superior strength of the exact separation procedure for MKP.

6 Conclusion

The local cuts approach was implemented for the 0–1 multidimensional knapsack problem. Based on LP solution of MKP, a reduced separation problem is solved exactly with the row generation algorithm. The valid inequality for the reduced MKP problem is then globally lifted to the original MKP problem. Computational experiments are carried out on a comprehensive set of MKP instances which demonstrate the superiority of the local cuts approach in terms of closed integrity gap. Further research is needed to reduce the computation time and investigate the possibility to combine the local cuts approach with a branch and bound algorithm or other metaheuristics.

References

1. Cho, Y.K., Moore, J.T., Hill, R.R., Reilly, C.H.: Exploiting empirical knowledge for bi-dimensional knapsack problem heuristics. Int. J. Ind. Syst. Eng. **3**(5), 530–548 (2008)
2. Chvátal, V., Cook, W., Espinoza, D.: Local cuts for mixed-integer programming. Math. Program. Comput. **5**(2), 171–200 (2013). doi:10.1007/s12532-013-0052-9
3. Chu, P.C., Beasley, J.E.: A genetic algorithm for the multidimensional knapsack problem. J. Heuristics **4**(1), 63–86 (1998)
4. Fréville, A.: The multidimensional 0–1 knapsack problem: an overview. Eur. J. Oper. Res. **155**(1), 1–21 (2004)
5. Fréville, A., Hanafi, S.: The multidimensional 0–1 knapsack problem—bounds and computational aspects. Ann. Oper. Res. **139**(1), 195–227 (2005)
6. Fukasawa, R., Goycoolea, M.: On the exact separation of mixed integer knapsack cuts. Math. Program. **128**(1), 19–41 (2011). doi:10.1007/s10107-009-0284-7
7. Gu, H.: Improving problem reduction for 0–1 multidimensional knapsack problems with valid inequalities. Comput. Oper. Res. **71**(C), 82–89 (2016). doi:10.1016/j.cor.2016.01.013
8. Gu, Z., Nemhauser, G.L., Savelsbergh, M.W.P.: Lifted cover inequalities for 0–1 integer programs: computation. INFORMS J. Comput. **10**(4), 427–437 (1998)
9. Hill, R.R., Moore, J.T., Hiremath, C., Cho, Y.K.: Test problem generation of binary knapsack problem variants and the implications of their use. Int. J. Oper. Quant. Manag. **18**(2), 105–128 (2011)
10. Kaparis, K., Letchford, A.N.: Local and global lifted cover inequalities for the 0–1 multidimensional knapsack problem. Eur. J. Oper. Res. **186**(1), 91–103 (2008)
11. Kaparis, K., Letchford, A.N.: Separation algorithms for 0–1 knapsack polytopes. Math. Program. **124**(1–2), 69–91 (2010). doi:10.1007/s10107-010-0359-5
12. Kellerer, H., Pferschy, U., Pisinger, D.: Knapsack Problems. Springer (2004)
13. Vasilyev, I., Boccia, M., Hanafi, S.: An implementation of exact knapsack separation. J. Glob. Optim. 1–24 (2015). doi:10.1007/s10898-015-0294-3
14. Wilbaut, C., Hanafi, S., Salhi, S.: A survey of effective heuristics and their application to a variety of knapsack problems. IMA J. Manag. Math. **19**(3), 227–244 (2008)
15. Wolsey, L.A., Nemhauser, G.L.: Integer and Combinatorial Optimization. Wiley Series in Discrete Mathematics and Optimization. Wiley (1999)