Journal of the Operations Research Society of Japan
Vol. 45, No. 3, September 2002

A SHORT NOTE ON THE REDUCIBILITY OF THE COLLAPSING KNAPSACK PROBLEM

Hiroshi Iida
Otaru University of Commerce

Takeaki Uno
National Institute of Informatics

(Received October 15, 2001)

Abstract This paper deals with the collapsing knapsack problem. In the literature, to solve the problem, a method incorporating a reduction from the problem to the 0–1 knapsack problem has been proposed. In this paper we show an alternative reduction which produces coefficients smaller than those by the previous. The improvement makes it possible to solve the resulting 0–1 knapsack problem faster than the previous. On our estimation in a case, the efficiency attains up to 150 times. We also show that the coefficients produced will be the smallest possible.

1. Introduction

In the classical 0–1 knapsack problem (KP), where items and a knapsack of a certain capacity are given, we pack the items into the knapsack so that the total profit of the packed items is maximized without exceeding the capacity. In the KP, the capacity of the knapsack is constant, while Posner and Guignard [4] introduced a more complicated problem with a nonconstant capacity, named Collapsing 0–1 Knapsack Problem (CKP). In the CKP, the knapsack will collapse according to the number of packed items. For instance, each item is an antique, and should be covered with something strong respectively when packed. Then the larger the number of packed items, the smaller the capacity of the knapsack, due to the strong covering each item. For more applications, see [4]. The CKP is formally stated as follows:

$$\begin{aligned} \text{(CKP)} \quad & \text{maximize} && \sum_{i\in N} c_i x_i \\ & \text{subject to} && \sum_{i\in N} a_i x_i \le b\left(\sum_{i\in N} x_i\right) \\ & && x_i \in \{0,1\},\ i \in N, \end{aligned} \tag{1.1}$$

where $N := \{1, 2, \ldots, n\}$ and each $i \in N$ indicates an item. The 0–1 variable x_i corresponds to the selection of item i, that is, $x_i = 1(\text{select})/x_i = 0(\text{no select})$. The numbers c_i and a_i associated with each item i will be called the profit and weight of item i, respectively. Throughout this paper we assume that both the profit c_i and weight a_i of any item i are given positive integers. Also, $b(\cdot)$ represents a capacity of the knapsack, and is a given monotone nonincreasing function on the discrete domain N as $b(1) \ge b(2) \ge \cdots \ge b(n)$. In the case where $b(\cdot)$ is constant, the CKP is reduced to KP. The CKP thus includes KP as a special case, and is $\mathcal{NP}$-hard.

In the literature several algorithms for CKP have been proposed, e.g. Fayard and Plateau [1], and Pferschy et al [3]. In particular, Pferschy et al [3] proposed two simple but efficient algorithms, in which our interest is in the first one. It incorporates a reduction which produces a KP equivalent to given CKP, and solves the resulting instance with an algorithm for KP. On the other hand, the second algorithm is tailored for CKP.

In fact the first algorithm is outperformed by the second, one reason for which is that very large coefficients appear by the reduction, which makes the resulting instance hard to solve. Still the reduction scheme seems attractive, because the KP is $\mathcal{NP}$-hard whereas not only it is solvable in pseudopolynomial time with dynamic programming but also there exist several efficient approaches to solve KP, e.g. a tight upper bound obtained by LP-relaxation or others, fixing 0–1 variables, and introducing core, etc. For a comprehensive overview of recent studies on KP, see Martello et al [2].

In the next section we show another reduction which produces coefficients smaller than those in [3]. We also show that the coefficients produced will be the smallest possible.

2. Another Reduction

Based on the CKP (1.1), the reduction proposed in [3] constructs KP with $2n$ items of weights $\alpha_1, \dots, \alpha_{2n}$ and profits $\gamma_1, \dots, \gamma_{2n}$. The coefficients are defined as follows:

$$\alpha_i = \begin{cases} a_i + A & \text{for } i = 1, \dots, n \\ (4n - i)A - b(i - n), & \text{for } i = n+1, \dots, 2n \end{cases}$$
$$\gamma_i = \begin{cases} c_i + C & \text{for } i = 1, \dots, n \\ (3n + 1 - i)C, & \text{for } i = n+1, \dots, 2n, \end{cases}$$

where $A = \sum_{i \in N} a_i$ and $C = \sum_{i \in N} c_i$. Then the resultant KP, which is called SKP, is stated as follows:

$$\begin{array}{lll} \text{(SKP)} & \text{maximize} & \sum_{i=1}^{2n} \gamma_i x_i \\ & \text{subject to} & \sum_{i=1}^{2n} \alpha_i x_i \le B \\ & & x_i \in \{0,1\}, \ i = 1, \dots, 2n, \end{array}$$

where the capacity is defined as $B = 3nA$. Following the terminology in [3] we hereafter call an item with index in N *small item*, and with index in $\{n+1, \dots, 2n\}$ *large item*. The following validates the equivalency between CKP and SKP:

Theorem (Pferschy et al, 1997). The instance of CKP has a feasible solution with objective value V if and only if the instance of SKP has a feasible solution with objective value $V + (2n+1)C$.

Here, with respect to a solution (0–1 n-vector) x of CKP with $\sum_{i \in N} x_i = j$, we define a solution corresponding to x in SKP as a 0–1 $2n$-vector the elements of which are x for small items and the remaining n of ($x_{n+j} = 1$; 0 otherwise) for large items. The reason for this definition is that packed items of which the total profit is maximized in SKP without exceeding the capacity comprise one large item $n+i$ and just i small items, and furthermore the total profit of the i small items calculated in given CKP results in being maximized, the essence of which could be summarized as follows:

1. A solution x is feasible in given CKP if and only if a solution corresponding to x in SKP is feasible.
2. At most one large item can be packed into the knapsack.
3. The total weight of large item $n + i$ and any combination of more than i small items exceeds the capacity.

4. The total profit of large item $n+i$ and any combination of less than i small items does not achieve an optimal value.
5. At least one large item should be packed to achieve an optimal value. In other words, the total profit of all small items is less than an optimal value.

Namely, if SKP satisfies these five points then a solution of given CKP which is corresponding to an optimal solution of the SKP is also optimal in the CKP. In what follows we re-construct the coefficients of SKP according to these five points: The first three are concerned with α_i's and B, and the remaining two are γ_i's. Since the following reduction does not take advantage of the monotonicity of $b(\cdot)$ as same as the previous, it is also valid for any capacity function $b(\cdot)$. Throughout this paper, without loss of generality, we assume $b(i) \geq 0$ for any i; otherwise we replace the negative $b(i)$ with 0.

First, let weights be as follows:

$$\begin{cases} \alpha_i &= a_i + A & \text{for } i = 1, \dots, n \\ \alpha_{n+i} &= (t-i)A + s - b(i), & \text{for } i = 1, \dots, n, \end{cases}$$

and x be a feasible solution of CKP with $\sum_{i \in N} x_i = j$.

On the first point above: In order that a solution corresponding to x in SKP is also feasible, we have $\alpha_{n+j} + \sum_{i \in N} a_i x_i + jA \leq B$. Then, $B := tA + s$. Conversely, a capacity $B - \alpha_{n+i} - iA = b(i)$ should remain for i items in CKP, which also implies $B = tA + s$. In the following we assume that all weights of CKP are sorted in nondescending order such that $a_1 \leq a_2 \leq \cdots \leq a_n$. In addition we define $\text{amin}(i) := \sum_{j=1}^{i} a_j$.

On the second point above: First, any large item is available since, assuming $A \geq 0$, we have

$$\alpha_{n+i} = B - iA - b(i) \leq B, \quad \text{for } i = 1, 2, \dots, n.$$

Second, we assume $s > b(i) + b(j)$ for any $1 \leq i < j \leq n$. In the case where $A > 0$,

$$\begin{aligned} \alpha_{n+i} + \alpha_{n+j} &= (t-i)A + s - b(i) + (t-j)A + s - b(j) \\ &> (2t - i - j)A + s \\ &\geq (2t - 2n + 1)A + s \geq tA + s = B. \end{aligned}$$

By the last inequality we have $t := 2n - 1$ as the smallest possible. Also, $s := \max_{i \neq j}\{b(i) + b(j)\} + 1$; otherwise $A = 0$, it can be confirmed that, with the s determined, the inequality of $\alpha_{n+i} + \alpha_{n+j} > B$ is still valid under $A = 0$. Last, the s and t obtained implies $\alpha_{n+i} > 0$ for all large items.

On the third point above, because $\alpha_i > 0$ for any small item by the assumptions of $a_i > 0$ for any i and $A \geq 0$, the following for $i = 1, 2, \dots, n-1$ validates the point:

$$\begin{aligned} &\alpha_{n+i} + \text{amin}(i+1) + (i+1)A \\ &= (t-i)A + s - b(i) + \text{amin}(i+1) + (i+1)A \\ &> tA + s = B. \end{aligned} \tag{2.1}$$

Then we have $A > b(i) - \text{amin}(i+1)$. Therefore, A is determined as follows:

$$A := \max_{i=1,\dots,n-1}\{b(i) - \text{amin}(i+1)\} + 1.$$

In the case where this A is negative against the assumption $A \geq 0$, we replace the value with 0. Incidentally, if the A calculated is negative then it follows that $\text{amin}(i+1) > b(i)$

for $i = 1, 2, \ldots, n-1$, which is the same as the condition (2.1) with $A = 0$. To summarize,

$$\begin{aligned}
\alpha_i &= \begin{cases} a_i + A & \text{for } i = 1, \ldots, n \\ (3n-1-i)A - b(i-n) + \max_{j \neq k}\{b(j) + b(k)\} + 1, & \text{for } i = n+1, \ldots, 2n \end{cases} \\
A &= \max\left\{\max_{i=1,\ldots,n-1}\{b(i) - \text{amin}(i+1)\} + 1, 0\right\} \\
B &= (2n-1)A + \max_{j \neq k}\{b(j) + b(k)\} + 1.
\end{aligned}$$

Next, let profits be as follows:

$$\begin{cases} \gamma_i &= c_i + C \quad \text{for } i = 1, \ldots, n \\ \gamma_{n+i} &= (t-i)C + s, \quad \text{for } i = 1, \ldots, n. \end{cases}$$

Here we assume that all profits of CKP are sorted in nondescending order such that $c_1 \leq c_2 \leq \cdots \leq c_n$. In addition we define $\text{cmax}(i) := \sum_{j=n-i+1}^{n} c_j$. Moreover we employ a feasible solution of CKP, say x', and let cmin be a profit given by x', i.e. $\sum_{i \in N} c_i x'_i$. Assuming $\sum_{i \in N} x'_i = j$ we have a profit given by a feasible solution corresponding to x' in SKP as $\gamma_{n+j} + \text{cmin} + jC = tC + \text{cmin} + s$.

For the fourth point above, under an assumption $C \geq 0$ to have $\gamma_i > 0$ for any small item, it is sufficient to ensure that, for $i = 1, 2, \ldots, n$,

$$(t-i)C + s + \text{cmax}(i-1) + (i-1)C < tC + \text{cmin} + s.$$

Then we have $C > \text{cmax}(i-1) - \text{cmin}$. Thus,

$$\begin{aligned}
C &:= \max\{\max_{i=1,\ldots,n} \text{cmax}(i-1) - \text{cmin} + 1, 0\} \\
&= \max\{\text{cmax}(n-1) - \text{cmin} + 1, 0\}.
\end{aligned}$$

For the fifth point above, it is sufficient to ensure that

$$\text{cmax}(n) + nC < tC + \text{cmin} + s. \tag{2.2}$$

Here we assume $s \geq c_1$. In the case where $C > 0$, we have

$$\begin{aligned}
\text{cmax}(n) + nC &\leq (n+1)C + \text{cmax}(n) - \text{cmax}(n-1) + \text{cmin} - 1 \\
&= (n+1)C + \text{cmin} + c_1 - 1 \\
&< (n+1)C + \text{cmin} + s.
\end{aligned}$$

Hence $t := n+1$ is the smallest possible. Also $s := c_1$; otherwise $C = 0$, it can be confirmed that, with the s determined and an inequality of $\text{cmax}(n-1) < \text{cmin}$ obtained from the C defined above, the condition (2.2) is still valid under $C = 0$. Summarizing,

$$\begin{aligned}
\gamma_i &= \begin{cases} c_i + C & \text{for } i = 1, \ldots, n \\ (2n+1-i)C + c_1, & \text{for } i = n+1, \ldots, 2n \end{cases} \\
C &= \max\{\text{cmax}(n-1) - \text{cmin} + 1, 0\} \\
\text{cmin} &= \sum_{i \in N} c_i x'_i, \ x' \text{ is a feasible solution of CKP.}
\end{aligned}$$

As observed above, the magnitude of γ_i's is in inverse proportion to the one of cmin. To obtain x' by which a fairly large profit is given, ordinary greedy heuristic will be of use.

Example. Consider an instance of CKP with $n = 3$; items are given as $\{(a_i, c_i)\} = \{(2,2),(2,3),(2,4)\}$, and $b(1) = 5, b(2) = 4, b(3) = 3$. Then $A = 2$, and we have $C = 1$, employing a feasible solution $x' = (0,1,1)$ with cmin $= 7$. The coefficients of SKP corresponding to the instance are as follows:

i	1	2	3	4	5	6
a_i	2	2	2			
c_i	2	3	4			
α_i	4	4	4	13	12	11
γ_i	3	4	5	5	4	3
B				20		

The optimal value of this SKP is 13 gained by a solution $(0,1,1,0,1,0)$. Extracting a part for small items from it we have a solution $(0,1,1)$, which is optimal to the given CKP as expected.

We would like to add that, as observed in Example, large item $2n$ provided for all items being packed in CKP is redundant, provided $b(n) < \sum_{i \in N} a_i$. In general, on an instance of CKP, we would obtain $m := \max\{i \mid \sum_{j=1}^{i} a_j \le b(i)\}$ less than n, i.e. more than m items cannot be packed. In this case we can customize coefficients of SKP. Concretely,

$$\begin{aligned}
\alpha_i &= \begin{cases} a_i + A & \text{for } i = 1, \dots, n \\ (2m + n - 1 - i)A - b(i-n) + b(1) + b(2) + 1, & \text{for } i = n+1, \dots, n+m \end{cases} \\
A &= \max\{b(1) - a_1 - a_2 + 1, 0\} \\
B &= (2m-1)A + b(1) + b(2) + 1 \\
\gamma_i &= \begin{cases} c_i + C & \text{for } i = 1, \dots, n \\ (2n + 1 - i)C + \sum_{j=1}^{n-m+1} c_j, & \text{for } i = n+1, \dots, n+m \end{cases} \\
C &= \max\{\text{cmax}(m-1) - \text{cmin} + 1, 0\}.
\end{aligned}$$

Applying, to the CKP in Example, the above with the same $x' = (0,1,1)$ as the previous we have the following SKP:

i	1	2	3	4	5
α_i	4	4	4	9	8
γ_i	2	3	4	5	5
B			16		

Finally, based on the reduction above, we will roughly estimate how many times the first algorithm in [3] improves. Here we employ the examined data provided in [3], especially the last one in Table 4 (large-sized problems), where $n = 1000$, weights a_i's are randomly distributed in [1, 1000], and $b(i)$'s are bounded by 50000. In the case where $b(i) = 0$ set for $i > 100$, the average solution time of the first algorithm is 938.18 seconds.

The first algorithm solves SKP by dynamic programming, and its time bound is $O(nB)$, where $B = 3nA$ in [3]. On the reduction above, first, the coefficient 3 is about one third smaller than the previous. Next we have $m \le 100$ in the data above, then it is ten times smaller than n or less. Last, the previous $A = \sum_{i \in N} a_i$ would be estimated at about 500000 in the data above, while our A is less than 50000; thus it is also ten times smaller. Totally, the first algorithm with our reduction will be performed in $938.18 \times (2/300) = 6.25$ seconds. Considering that the average solution time of the second algorithm is 16.10 seconds in Table 4, we may conclude that our reduction could make the first algorithm a promising competitor to the second.

References

[1] D. Fayard and G. Plateau: An exact algorithm for the 0-1 collapsing knapsack problem. *Discrete Applied Mathematics*, **49** (1994) 175–187.

[2] S. Martello, D. Pisinger, and P. Toth: New trends in exact algorithms for the 0–1 knapsack problem. *European Journal of Operational Research*, **123** (2000) 325–332.

[3] U. Pferschy, D. Pisinger, and G.J. Woeginger: Simple but efficient approaches for the collapsing knapsack problem. *Discrete Applied Mathematics*, **77** (1997) 271–280.

[4] M.E. Posner and M. Guignard: The collapsing 0–1 knapsack problem. *Mathematical Programming*, **15** (1978) 155–161.

Hiroshi Iida
Dept. of Information and Management Science,
Otaru University of Commerce, Otaru 047-8501, Japan.
E-mail: oggi@res.otaru-uc.ac.jp.