How to Solve the Collapsing Subset-Sum Problem

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

Abstract

In this short paper we present a new type of problem which we shall call collapsing subset-sum problem, and we also present a promising algorithm which could make it possible to solve the problem efficiently. Computational experiments are included.

Keywords: collapsing 0-1 knapsack problem; subset-sum problem; strongly correlated 0-1 knapsack problem

1 Introduction

In the classical 0-1 Knapsack Problem (KP) the capacity is constant, however, there exists a more complicated problem that has a nonconstant capacity. The Collapsing 0-1 Knapsack Problem (CKP) is such a type of non-linear knapsack problem, and is introduced by Posner & Guignard [4]. The CKP is formulated as follows:

$$\begin{aligned} &\text{maximize} \quad \sum_{j=1}^{n} p_j x_j \\ &\text{subject to} \quad \sum_{j=1}^{n} w_j x_j \le b\left(\sum_{j=1}^{n} x_j\right) \\ &\qquad\qquad x_j \in \{0, 1\},\ j = 1, 2, \dots, n, \end{aligned} \tag{1}$$

where both the profit p_j and weight w_j of any item j are given positive integers, and the $b(\cdot)$ on the right-hand side of the constraint is a given monotone nonincreasing function on the discrete domain $\{1, 2, \ldots, n\}$. Without loss of generality we will assume that $w_j \leq b(1)$ for any j, and that $\sum_j w_j > b(n)$. Also, we call the n variables of x_j '0-1 variables.' Clearly, the CKP is an extension of KP and is *NP*-hard as implied. The applications of CKP are also mentioned in [4].

Here we will consider applying an additional constraint such that $p_j = w_j$ for all j to (1) as same as producing the Subset-Sum Problem (SSP) from KP. Then, we have

$$\begin{aligned} &\text{maximize} && \sum_{j=1}^{n} w_j x_j \\ &\text{subject to} && \sum_{j=1}^{n} w_j x_j \leq b\left(\sum_{j=1}^{n} x_j\right) \\ & && x_j \in \{0, 1\}, \ j = 1, 2, \ldots, n. \end{aligned} \tag{2}$$

We hereby call the problem (2) *Collapsing Subset-Sum Problem* (CSSP). It is *NP*-hard as an extension of SSP.

Recently, Fayard & Plateau [1] have proposed an algorithm, named FPCK90 or FP for short, to solve CKP. Since the CSSP is a special case of CKP, FP is applicable to CSSP without modification. However it would seem that the performance of the algorithm depends on whether profit-to-weight ratios (p_j/w_j, efficiency) are widely distributed or not. Therefore, FP does not seem promising enough to solve CSSP. More precisely, FP could not fix so many 0-1 variables of CSSP by its size reduction tests, because employing the outer-linearization of FP in order to fix the 0-1 variables is identical to fixing those of the Strongly Correlated 0-1 Knapsack Problem (SCKP) with a negative fixed-charge. To the best of our knowledge it should be a hard issue.

Now we need an algorithm which could cope with not only the special structure of data instances that the profit-to-weight ratio is constant but also the monotone nonincreasing capacity. In the next section we present a promising algorithm to solve CSSP. In Section 3 computational experiments are presented to confirm our view. The last section is devoted to conclusion.

2. An algorithm for solving CSSP

Before presenting the detail, we introduce some definitions: Let $N := \{1, 2, \ldots, n\}$ and let J be a subset of N. Then, the cardinality of J and the sum of w_j for all $j \in J$ will be called the *length* and *weight* of J respectively. Sometimes we call a subset of N *solution*.

Truth to tell, the promising algorithm is a slightly modified LSSCOR which is an algorithm proposed by Pandit & Ravi Kumar [3] to solve SCKP. The algorithm LSSCOR transforms given SCKP to the equivalent that is SSP with an additional constraint:

$$\begin{aligned} &\text{maximize} && \sum_{j=1}^{n} p_j x_j \\ &\text{subject to} && \sum_{j=1}^{n} p_j x_j \leq c + \beta k, \quad \beta = \sum_{j=1}^{n} x_j \\ & && x_j \in \{0, 1\}, \; j \in N, \end{aligned} \tag{3}$$

where $p_j = w_j + k$ for any $j \in N$. If the fixed-charge k is negative, the right-hand side of the inequality, $c + \beta k$, is a monotone nonincreasing function of the number of 0–1 variables equal to one. This means that we can regard (3) as an instance of CSSP.

Conversely LSSCOR superseded $\mathrm{MCL}(\beta) := c + \beta k$ by $b(\beta)$ is applicable to CSSP, because LSSCOR does not depend on the linearity of $\mathrm{MCL}(\beta)$. Moreover, LSSCOR does not depend on the variety of profit-to-weight ratios, because it was developed to solve SCKP on which the profit-to-weight ratios are not distributed widely. Consequently the rough algorithmic sketch of LSSCOR utilized to solve CSSP is as follows:

Sort all weights such that $w_1 \geq w_2 \geq \cdots \geq w_n$;

LB := $\max_i \{ \sum_{j=1}^{i} w_j \leq b(i) \}$; /* LB $\geq$ 1 */

z^* := the weight of $\{1, 2, \ldots, \text{LB}\}$; /* initial incumbent value */

UB := $\max_i \{ \sum_{j=n-i+1}^{n} w_j \leq b(i) \}$;

i := LB + 1;

while $i \leq$ UB and $z^* < b(i)$ **do** /* main loop */

  Discard any w_j, provided $w_j > b(i) - \sum_{l=n-i+2}^{n} w_l$;

  Replace z^* with an improved one (if found)
    while enumerating all subsets of N with length i;

  $i := i + 1$

done;

The solution which gives z^* is optimal.

In the sketch, LB indicates a lower bound on the length of a solution. In addition, the subset $\{1, 2, \ldots, \text{LB}\}$ gives maximum weight among any subset of N the length of which is less than or equal to LB. Thus it is enough to make the main loop start at i = LB+1. Conversely, UB indicates an upper bound. In fact, it capitalizes on the nonincreasing capacity that the counter 'i' proceeds to UB in the main loop.

Also, the manner of enumerating the subsets of N with length i is the same as the one of LSSCOR. A remark is that if we find a solution the weight of which is equal to $b(i)$ while enumerating the subsets with length i, we can terminate the processing to optimality because we have found an optimal solution. It should be pointed out that there is a sense in which

SSP is easier than KP. Namely, on SSP, a solution the weight of which is equal to the capacity is optimal, while such a solution is not always so on KP.

3 Computational experiments

In this section we present computational experiments. The data instances provided for the experiments have been taken from Series 10–12 in [1], which are prescribed by the first row in Table 1: The m indicates that $m := \max_{i \in N} \{b(i) > 0\}$. The $b(\cdot)$ indicates the upper bound of capacities, that is, the nonnegative capacities $\{b(1), b(2), \ldots, b(m)\}$ are randomly generated in the range $[1, b(\cdot)]$ in descending order. The weights are randomly generated in the range [1, 1000] in all cases. In the case where $b(1) < 1000$ however we have adopted a new range $[1, b(1) - 1]$ for the weights.

In place of LSSCOR we here use the algorithm XISC proposed by Iida [2], which has the same facility as LSSCOR, solving SCKP efficiently. For the sake of solving CSSP, XISC has been slightly modified according to the sketch in the preceding section. The algorithm has been implemented in C and the data instances have been solved on SPARCstation-5. In Table 1, each column for XISC reports the average computation time of 100 data instances, expressed in seconds.

Table 1: the performance of XISC

n	$b(\cdot)$	m	XISC (sec.)
1000	1000	100	0.01000
		500	0.01000
	10000	100	0.01000
		500	0.01017
	50000	100	0.01000
		500	0.00967

4 Conclusion

As we have seen it so for, the collapsing 0-1 knapsack problem with an additional constraint that the profit is equal to the weight on all items can be solved in a fraction of a second. Although we do not know whether the constraint is meaningful or not in real-life, it would not at least be meaningless to make a suggestion that an intractable problem could be solved in reasonable computation time. Also it should be pointed out that CSSP can be solved very efficiently compared with CKP (see [1]), in contrast to that SSP is harder than KP owing to the constant profit-to-weight ratio.

References

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

[2] H. Iida, "A simple branch-and-bound approach for the strongly correlated knapsack problem," *Economic Review* **48**(2·3), 353-370, Otaru University of Commerce (Jan. 1998).

[3] S.N.N. Pandit and M. Ravi Kumar, "A lexicographic search for strongly correlated 0-1 knapsack problems," *Opsearch* **30**(2), 97-116 (1993).

[4] M.E. Posner and M. Guignard, "The collapsing 0-1 knapsack problem," *Math. Programming* **15**(2), 155-161 (1978).