A Hybrid approach for zero-one Multiple-Choice Knapsack Problem with Setup

Yassine Adouani[1], Malek Masmoudi[2], Iyad Alghoul[3], Bassem Jarboui[3]

[1]Laboratory of Modeling and Optimization for Decisional, Industrial and Logistic Systems (MODILS), Faculty of Economics and Management Sciences of Sfax, University of Sfax, Sfax, Tunisia

[2]University of Lyon, Saint Etienne, France; University of Saint Etienne, Jean Monnet, Saint-Etienne, France; LASPI, IUT de Roanne, Roanne, France

[3]Emirates College of Technology, Abu Dhabi, United Arab Emirates

[*]Corresponding author:

Yassine Adouani (Tel: +21652160714; e-mail: adouaniyassine@gmail.com)

Coauthors:

Malek Masmoudi: malek.masmoudi@univ-st-etienne.fr

Iyad Alghoul: iyad.alghoul@ect.ac.ae

Bassem Jarboui: bassem_jarboui@yahoo.fr

Abstract: This paper presents a hybrid approach to solve the Multiple-Choice Knapsack problem with Setup (MCKS), a variant of 0-1 knapsack problem with setup. This paper provides a cooperative approach that combines iterated local search (ILS) technique with IP. We considered three local search techniques to assign the classes to knapsack, and then if the assignment is identified to be promising by comparing its result to the upper bound, we applied the IP to select the items in knapsack. For the numerical experiment, we generated different instances for MCKS. In the experimental setting, we compared our cooperative approach to IP. Experimental results clearly showed the efficiency and effectiveness of our cooperative approach with a considerable gap of the objective function value for the MCKS (-0.04%) in a considerable shorter computation time (12 s vs. 2310 s).
Keywords: Knapsack problem; Setup; Cooperative approach; Integer programming

1. Introduction

The 0-1 Multiple-choice Knapsack Problem with Setup (MCKS) is described as a knapsack problem with additional the setup variables discounted both in the objective function and in the constraint. Practical applications of the MCKS may be seen in production scheduling problems involving setups and machine preferences. A case study of KPS is provided in Della et al. [8]. The case study is originated within the smart-home paradigm where the goal of an efficient management of the buildings energy consumptions is a strong component. To extend the KPS to MCKS, we consider that items from the same family (or class) could be processed in multiple periods. Another application of MCKS may be seen in regional project selection in multiple periods for an organization (country or company) which has a fixed budget to invest in a number of projects in multiple areas which can be done in multiple periods [1].

The MCKS is NP-hard problem, since it is a generalization of the standard KP. MCKS reduces to a KP when considering one class, and no setup variables. So far, MCKS few considered in the literature. Yang provided an exact method based on a branch and bound for the MCKS [1], but it has no availability of benchmark instances in the literature. Note, MCKS has similarities to several existing problems in the literature such as the multiple choice knapsack problems (MCK) when ignoring the setup variables [9], the KPS is a particular case of MCKS, when the number of period is equal to one (T=1) [2, 3, 8], etc.

To deal with the different variants of KP, exact techniques are introduced in the literature such as branch and bound algorithm for KPS [1], Chebil and Khemakhem [2] provided an improved dynamic programming algorithm for KPS. The hybridization technique between exact and metaheuristics approaches have been performed by many researchers during the last few decades. This technique provides interesting results as they take advantages of both types of methods [4]. A classification of algorithms combining local search techniques and exact methods is given in

Dumitrescu and Stutzle [5]. The focus is particularly on the so called cooperative algorithms using exact methods to strengthen local search techniques [6, 7].

2. Problem description

We propose the following mathematical formulation for the MCKS (see Yang [1]).

Using integer programming (IP) to solve MCKS shows its limitation due to the complexity of the problem. Thus, we decided to look for alternative approaches. We chose the cooperative approach combining local search techniques and exact method. We explain our new approach in the next section.

3. Cooperative approach for MCKS

The main idea of our cooperative approach was to decompose the original problem into a first problem to assign classes to knapsack (determine y_{it}) using an iterated local search (ILS) approach. This approach allowed transformation of MCKS into classical KP. Then in the second problem, if the assignment is identified to be promising by comparing its result to the upper bound, we applied the IP to select the items in the KP. The IP computation time is very short and thus determines the values of x_{ijt}. Note these found values of y_{it} and x_{ijt} yield a feasible solution to MCKS. For the approach effectiveness and efficiency, we developed a construction heuristic called reduction-based heuristic (see, For example, Appendix A) that gives a good initial solution, and it provides a lower bound. This was done in order to accelerate the convergence by exploring good candidates in the search space. Three procedures *SWAP&IP*, *INSERT&IP* and *PERTURB&IP* were considered within our cooperative approach. We obtained them by combining local search techniques *SWAP*, *INSERT* and *PERTURB* with IP.

4. Computational results

For computation, our approach was implemented and run using C language and CPLEX 12.7 solver on a 2.4 GHZ intel B960 computer with 4 GB of memory. Due to the unavailability of benchmark instances in the literature, we tested our cooperative approach *ILS&IP* on a set of randomly generated instances of MCKS with a total number of periods T in $\{5, 10, 15, 20\}$, total number of classes N in $\{10, 20, 30\}$, and total number of items n_i for each class i in $[40,60]$, $[60,90]$ and $[90,110]$ (Available at https://goo.gl/y4R7HW). We generated *360* instances in total: *10* instances for each combination (T, N). We designed a random generation scheme, as presented in [2], where:

- $f_{it} = -\sum_{j=1}^{n_i} c_{ijt} * e$, e is selected with a uniform distribution in[0.15,0.25]
- $c_{ijt} = a_{ij} + e_1$, $e_1$is selected with a uniform distribution in[0,10]

The Gap report the standard deviation between IP and ILS&IP that is calculated as follows: $\text{Gap}(\%) = 100 * \left(\frac{\text{IP}_{\text{sol}} - \text{ILS \& } IP_{\text{sol}}}{\text{IP}_{\text{sol}}}\right)$. `

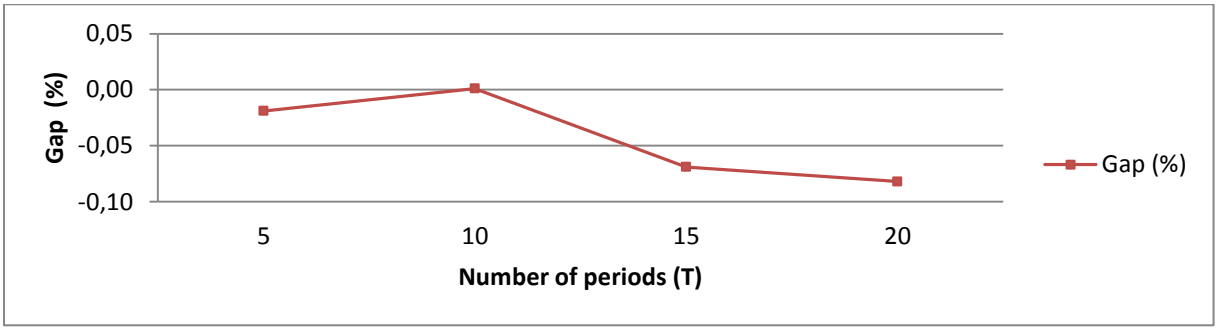

Figure 1: Gap (%) between the ILS&IP and the IP for each period

Figure 1 shows that ILS&IP outperforms the IP with a gap on average equal to -0.042%. In details, the gap on average is about -0.019% for $T = 5$, 0.001% for $T = 10$, -0.069% for $T = 15$, and -0.082% for $T = 20$. For more detailed results, we note that the ILS&IP provides a solution equal to the one provided by the IP for 203 instances and provides better solutions that the IP for 126 instances. The IP finds the optimal solutions for 166 instances, slightly outperforms the ILS&IP for 31 instances, and for the remaining it terminates with error: exceeds the capacity of RAM memory or exceeds the CPU time limit. In fact, the number of times that the IP terminates with exceeding the capacity of RAM or exceeding the time limit increases from 20 with $T = 5$ to 70 with $T = 20$.

The CPU on average for the ILS&IP is about 12 seconds which is very low in comparison to the average of CPU for IP that is equal to 2310 seconds. A complete and detailed version of the results is available at https://goo.gl/bPLMho.

5. Conclusion

In this paper, we introduced MCKS, and proposed a new cooperative approach that combines ILS and IP. Our cooperative approach denoted ILS&IP is tested on a wide set of instances that are generated for MCKS problems. The experimental results showed that ILS&IP produced good quality (optimal and near-optimal solutions) solutions in a short amount of time. For future work, we expect to generalize our cooperative approach to deal with other variants of knapsack problem.

References

[1] Yang, Y. (2006). Knapsack problems with setup. Dissertation, Auburn University.

[2] Chebil, K., & Khemakhem, M. (2015). A dynamic programming algorithm for the Knapsack Problem with Setup. Computers and Operations Research, 64, 40-50.

[3] Khemakhem, M., & Chebil, K. (2016). A Tree Search Based Combination heuristic for the Knapsack Problem with Setup, Computers & Industrial Engineering, 99, 280-286.

[4] Plateau, G., & Elkihel, M. (1985). A hybrid method for the 0-1 knapsack problem. Operations Research, 49, 277-293.

[5] Dumitrescu, I., & Stutzle, T. (2003). Combinations of local search and exact algorithms. In G.L. Raidl et al, editor, Applications of evolutionary computation. Springer-Verlag, 2611, 211-223.

[6] Vasquez, M., & Hao, J. K. (2001).A hybrid approach for the 0-1 multidimensional knapsack problem. In Proceedings of the International Joint Conference on Artificial Intelligence, 328-333.

[7] Fernandes, S., & Lourenco, H. (2007). Hybrid combining local search heuristics with exact algorithms. In F. Almeida et al, editor, V Congreso Espanolsobre Metaheursticas, Algoritmos Evolutivos y Bioinspirados, 269-274.

[8] Della, C. F., Salassa, F., & Scatamacchia, R. (2017). An exact approach for the 0-1 knapsack problem with setups. Computers & Operations Research, 80, 61-67.

[9] Pisinger, D. (1995). A minimal algorithm for the multiple-choice knapsack problem European Journal of Operational Research 83, 394-410.