A Cooperative Learning Approach for the Quadratic Knapsack Problem

Eduardo Lalla-Ruiz[1], Eduardo Segredo[2,3], and Stefan Voß[1]

[1] Institute of Information Systems. University of Hamburg, Germany. {eduardo.lalla-ruiz, stefan.voss}@uni-hamburg.de
[2] School of Computing. Edinburgh Napier University, UK. e.segredo@napier.ac.uk
[3] Dpto. de Ingeniería Informática y de Sistemas. Universidad de La Laguna, Spain. esegredo@ull.edu.es

Abstract. The *Quadratic Knapsack Problem* (QKP) is a well-known optimization problem aimed to maximize a quadratic objective function subject to linear capacity constraints. It has several applications in different fields such as telecommunications, graph theory, logistics, hydrology and data allocation, among others. In this short paper, we propose the application of a novel population-based metaheuristic, which exploits the concepts of cooperation and communication along the search leading to a collective learning, to solve a wide range of well-known QKP instances.

1 Introduction

The *Quadratic Knapsack Problem* (QKP) is a knapsack problem introduced by Gallo *et al.* [4] that includes the relationship among the items within a quadratic objective function. Formally, in the QKP, we are given a set of items $N = \{1, ..., n\}$ where each item i has a given weight $w_i > 0$ and a $n \times n$ profit matrix B that indicates the benefit obtained when an item is packed with respect to itself and other items. In other words, if an item i is selected the profit obtained is equal to $b_{ii} + \sum_{j \in N \setminus \{i\}} b_{ij}$. We should note that matrix B is symmetric, i.e., $b_{ij} = b_{ji}$. The goal of the QKP is thus determining the items to be packed in the knapsack taking into account the weight capacity c such that the total profit of the items packed is maximized. The selection of the items is ruled by the binary variable x_i which is equal to 1 if item i is selected and 0 otherwise. The formal definition of the QKP is as follows.

$$\max z(QKP) = \sum_{i \in N} \sum_{j \in N} b_{ij} x_i x_j \tag{1}$$

$$\sum_{i \in N} w_i^k x_i \leq c \tag{2}$$

$$x_i \in \{0, 1\}, \quad i \in N \tag{3}$$

As can be deduced from the above formulation, if $b_{ij} = 0$ for $i \neq j$ then the QKP can be reduced to the *Knapsack Problem*. Moreover, the *Clique Problem* can be obtained as a particular case of the QKP, where the *Max Clique* can be solved by means of a QKP algorithm by using a binary search [1].

In this work, we propose the application of a novel population-based metaheuristic, called *Multi-leader Migrating Birds Optimization* (MMBO) [5]. This approach exploits the communication among a population of individuals during the search, thus enabling cooperative learning. We test our method for the latest benchmark suite proposed for the QKP [2]. As it is shown in Section 3, our proposal reports highly competitive results, in terms of the objective function value attained at the end of the runs, in comparison to the best-performing state-of-the-art approaches that can be found in the related literature.

2 Multi-leader Migrating Birds Optimization

For solving the QKP, we propose the use of MMBO (Algorithm 1), which is a decentralized cooperative search approach inspired by the flight formation of migratory birds. In MMBO, the population is denoted as $P = \{1, 2, ..., p\}$, where p is the number of individuals representing solutions of the optimization problem at hand. During the search, individuals are distributed in a line formation, i.e., $(1, 2, ..., p)$, where individual 1 is directly connected to individual 2, and individual 2 is connected to individuals 1 and 3, and so on. Based on that line formation, a relationship structure is established according to a given *relationship criterion*, for instance, in terms of the objective function value associated with each member in the population. By means of that criterion, the role among each pair of individuals is determined, i.e., which individual provides and which individual receives information during the search. Starting from each individual in P, k feasible neighbors are generated through a predefined neighborhood structure. In this work, two decision variables of a given individual are uniformly selected at random and their corresponding binary values are flipped in order to produce a novel neighbor. Depending on the relationship criterion and how information is shared among individuals, different roles arise:

- **Leader**. It is that individual with the best objective function value when compared to its adjacent individuals. Therefore, it does not receive information from any individual, but shares information, in the form of δ neighbors, with its adjacent individuals. Furthermore, starting from a leader, k neighbors are generated. Since the objective function value determines the position within the relationship structure, a leader is the fittest individual. The set of leaders is denoted as P_L.
- **Follower**. It is that individual which explores the search space considering its own information and the information received from the individuals in front of it within the relationship structure. It generates $k - \delta$ neighbors and receives δ neighbors from the adjacent individuals. The set of followers is denoted as P_F.
- **Independent**. It is that individual which is not included into any of the above categories because it has associated the same objective function value

than its adjacent individuals. Hence, it does not exchange information with any other individual, but generates k neighbors. The set of independent individuals is denoted as P_I.

Algorithm 1 Pseudocode of MMBO

```
Require: $p$, $K$, $k$, and $\delta$
 1: Initialize the population $P$, which consists of $p$ individuals generated at random
 2: while ($K$ neighbors have not been generated) do
 3:    Determine the interaction among the individuals of $P$ and establish the relation-
       ship structure
 4:    while (the stopping criterion associated with the relationship structure is not
       met) do
 5:       Generate $k$ neighbors starting from each individual included into $P_L \cup P_I$
 6:       Replace each individual included into $P_L \cup P_I$ by its fittest neighbor if the
          latter is fitter than the former
 7:       Replace each individual included into $P_I$ by its fittest neighbor
 8:       for all (individual $f \in P_F$) do
 9:          Generate $k - \delta$ neighbors starting from $f$
10:          Get the best unused $\delta$ neighbors from the previous individuals of $f$ in the
             relationship structure
11:          Replace $f$ by its fittest neighbor if the latter improves the former
12:       end for
13:    end while
14: end while
15: Return the fittest individual in $P$
```

3 Numerical Results

The proposed optimization technique has been implemented in Java and executed on a computer equipped with an Intel i7 CPU 3.5 GHz and 16 GB of RAM. By preliminary experiments, we identified the following parameters $p = 20$, $\delta = 1$, and $k = 5$ with a stopping criterion $K = |n|^2$ neighbors. The problem instances used are those proposed in [2] by following the guidelines of previous works.

Table 1 shows the computational results for instances with $n = 100$ items, which were generated by considering different density values (dst). The methods selected for comparison purposes are the best-performing ones in the related literature for the instances proposed in [2]. They are based on a *Dynamic Programming Heuristic* (DPH) and a *Non-Delayed Relax-and-Cut* (CSL) [3]. We should note that execution times are measured as integer values in [2]. Hence, some execution times were reported as 0 when the time invested was lower than 1 second. Therefore, we have reported the upper bound of the said times to avoid those cases. Furthermore, as it can be observed in the table, our approach is able to reduce considerably the execution time with respect to the best approach in terms of the objective function value (CSL). Finally, it is worth mentioning that MMBO reported a new best solution considering one of the instances ($id = 4$).

Table 1: Numerical results for instances of $n = 100$ items. The new best solution attained by MMBO is faced in **bold**

Instance		CSL		DPH		MMBO	
dst	id	obj	t(s)	obj	t(s)	obj	t(s)
25	1	53774	3	53757	2	53774	0.06
	2	7082	5	7076	1	7082	0.06
	3	60875	2	60875	1	60875	0.07
	4	18386	6	18386	2	18386	0.04
	5	43014	4	43014	1	43014	0.06
	6	50484	4	50484	1	50484	0.06
	7	21769	4	21769	2	21769	0.05
	8	30687	5	30687	1	30687	0.04
	9	28719	6	28719	1	28719	0.04
	10	5463	4	5421	2	5463	0.06
Average		32025.3	4.3	32018.8	1.4	32025.3	0.05

Instance		CSL		DPH		MMBO	
dst	id	obj	t(s)	obj	t(s)	obj	t(s)
50	1	34653	5	34653	1	34653	0.08
	2	43178	5	43169	1	43178	0.06
	3	46243	6	46243	2	46243	0.09
	4	48894	5	48992	1	**49030**	0.04
	5	41515	6	41515	1	41515	0.08
	6	71982	4	71982	2	71982	0.06
	7	69146	6	69177	1	69177	0.07
	8	83085	3	83085	1	83085	0.06
	9	9772	4	9772	2	9772	0.13
	10	62465	6	62407	1	62465	0.08
Average		51093.3	5	51099.5	1.3	51110	0.07

4 Conclusions

In this work, we have proposed a cooperative learning approach for the Quadratic Knapsack Problem. Due to its self-organization and cooperation dynamics, it allows individuals to learn along the search process in a collective way. The collaborative relationship structure enhances the diversification of the search as individuals can be distributed over the search space. At the same time, the intensification is addressed by the sum of efforts of the individuals belonging to the same group located at a particular region of the search space. It is remarkable from the computational experience that our algorithm reports high quality solutions in terms of the objective function value by investing shorter computational times with respect to state-of-the-art approaches. In this regard, we even attained a new best solution for one of the instances tested.

As future work, we aim to extend the results provided herein to analyze the different features provided by the approach to solve the QKP in more detail. We also aim to work on the assessment of our algorithm to deal with other problems.

References

1. A. Caprara, D. Pisinger, and P. Toth. Exact solution of the quadratic knapsack problem. *INFORMS Journal on Computing*, 11(2):125–137, 1999.
2. J. O. Cunha, L. Simonetti, and A. Lucena. Lagrangian heuristics for the quadratic knapsack problem. *Computational Optimization and Applications*, 63(1):97–120, 2016.
3. F. Fomeni and A. Letchford. A dynamic programming heuristic for the quadratic knapsack problem. *INFORMS Journal on Computing*, 26(1):173–182, 2013.
4. G. Gallo, P. L. Hammer, and B. Simeone. Quadratic knapsack problems. In M. W. Padberg, editor, *Combinatorial Optimization*, pages 132–149. Springer, Berlin, Heidelberg, 1980.
5. E. Lalla-Ruiz, J. de Armas, C. Expósito-Izquierdo, B. Melián-Batista, and J. M. Moreno-Vega. Multi-leader migrating birds optimisation: a novel nature-inspired metaheuristic for combinatorial problems. *International Journal of Bio-Inspired Computation*, 10(2):89–98, 2017.