Ant Colony Optimization for Multiple Knapsack Problem and Model Bias

Stefka Fidanova

IPP – BAS, Acad. G. Bonchev str. bl.25A, 1113 Sofia, Bulgaria
stefka@parallel.bas.bg

Abstract. The Ant Colony Optimization (ACO) algorithms are being applied successfully to a wide range of problems. ACO algorithms could be good alternatives to existing algorithms for hard combinatorial optimization problems (COPs). In this paper we investigate the influence of model bias in model-based search as ACO. We present the effect of two different pheromone models for ACO algorithm to tackle the Multiple Knapsack Problem (MKP). The MKP is a subset problem and can be seen as a general model for any kind of binary problems with positive coefficients. The results show the importance of the pheromone model to quality of the solutions.

1 Introduction

There are many NP-hard combinatorial optimization problems for which it is impractical to find an optimal solution. Among them is the MKP. For such problems the reasonable way is to look for algorithms that quickly produce near-optimal solutions. ACO [1, 4, 2] is a meta-heuristic procedure for quickly and efficiently obtaining high quality solutions to complex optimization problems [9]. The ACO algorithms were inspired by the observation of real ant colonies. Ants are social insects, that is, insects that live in colonies and whose behavior is directed more to the survival of the colony as a whole than to that of a single individual component of the colony. An important and interesting aspect of ant colonies is how ants can find the shortest path between food sources and their nest. ACO is the recently developed, population-based approach which has been successfully applied to several NP-hard COPs [5]. One of its main ideas is the indirect communication among the individuals of a colony of agents, called "artificial" ants, based on an analogy with trails of a chemical substance, called pheromones which real ants use for communication. The "artificial" pheromone trails are a kind of distributed numerical information which is modified by the ants to reflect their experience accumulated while solving a particular problem. When constructing a solution, at each step ants compute a set of feasible moves and select the best according to some probabilistic rules. The transition probability is based on the heuristic information and pheromone trail level of the move (how much the movement is used in the past). When we apply ACO algorithm to MKP different pheromone models are possible and the influence on the results is shown.

The rest of the paper is organized as follows: Section 2 describes the general framework for MKP as a COP. Section 3 outlines the implemented ACO algorithm applied to MKP with two different pheromone models. In Section 4 experimental results over test problems are shown. Finally some conclusions are drawn.

2 Formulation of the problem

The MKP has received wide attention from the operations research community [7, 3, 8], because it embraces many practical problems. Applications include resource allocation in distributed systems, capital budgeting and cutting stock problems. In addition, MKP can be seen as a general model for any kind of binary problems with positive coefficients [6]. They are optimization problems for which the solutions are sequences of 0 and 1 and the coefficients in objective function are positive and coefficients in constraints are positive or zero. The MKP can be thought as a resource allocation problem, where we have m resources (the knapsacks) and n objects and object j has a profit p_j. Each resource has its own budget c_j (knapsack capacity) and consumption r_{ij} of resource i by object j. We are interested in maximizing the sum of the profits, while working with a limited budget.

The MKP can be formulated as follows:

$$\max \sum_{j=1}^{n} p_j x_j$$

$$\text{subject to} \sum_{j=1}^{n} r_{ij} x_j \leq c_i \quad i = 1, \dots, m \tag{1}$$

$$x_j \in \{0, 1\} \quad j = 1, \dots, n$$

x_j is 1 if the object j is chosen and 0 otherwise.

There are m constraints in this problem, so MKP is also called m-dimensional knapsack problem. Let $I = \{1, \dots, m\}$ and $J = \{1, \dots, n\}$, with $c_i \geq 0$ for all $i \in I$. A well-stated MKP assumes that $p_j > 0$ and $r_{ij} \leq c_i \leq \sum_{j=1}^{n} r_{ij}$ for all $i \in I$ and $j \in J$. Note that the $[r_{ij}]_{m \times n}$ matrix and $[c_i]_m$ vector are both non-negative.

In the MKP we are not interested in solutions giving a particular order. Therefore a partial solution is represented by $S = \{i_1, i_2, \dots, i_j\}$ and the most recent elements incorporated to S, i_j need not be involved in the process for selecting the next element. Moreover, solutions for ordering problems have a fixed length as we search for a permutation of a known number of elements. Solutions for MKP, however, do not have a fixed length. We define the graph of the problem as follows: the nodes correspond to the items, the arcs fully connect nodes. Fully connected graph mens that after the object i we can choose the object j for every i and j if there are enough resources and object j is not chosen yet.

3 ACO Algorithm for MKP

Real ants foraging for food lay down quantities of pheromone (chemical cues) marking the path that they follow. An isolated ant moves essentially at random but an ant encountering a previously laid pheromone will detect it and decide to follow it with high probability and thereby reinforce it with a further quantity of pheromone. The repetition of the above mechanism represents the auto catalytic behavior of real ant colony where the more the ants follow a trail, the more attractive that trail becomes.

The ACO algorithm uses a colony of artificial ants that behave as co-operative agents in a mathematical space were they are allowed to search and reinforce pathways (solutions) in order to find the optimal ones. Solution that satisfies the constraints is feasible. After initialization of the pheromone trails, ants construct feasible solutions, starting from random nodes, then the pheromone trails are updated. At each step ants compute a set of feasible moves and select the best one (according to some probabilistic rules) to carry out the rest of the tour. The transition probability is based on the heuristic information and pheromone trail level of the move. The higher value of the pheromone and the heuristic information, the more profitable it is to select this move and resume the search. In the beginning, the initial pheromone level is set to a small positive constant value τ_0 and then ants update this value after completing the construction stage. ACO algorithms adopt different criteria to update the pheromone level. In our implementation we use Ant Colony System (ACS) [4] approach.

In ACS the pheromone updating stage consists of:

3.1 Local Update Stage

While ants build their solution, at the same time they locally update the pheromone level of the visited paths by applying the local update rule as follows:

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \rho\tau_0 \tag{2}$$

Where ρ is a persistence of the trail and the term $(1-\rho)$ can be interpreted as trail evaporation.

The aim of the local updating rule is to make better use of the pheromone information by dynamically changing the desirability of edges. Using this rule, ants will search in wide neighborhood around the best previous solution. As shown in the formula, the pheromone level on the paths is highly related to the value of evaporation parameter ρ. The pheromone level will be reduced and this will reduce the chance that the other ants will select the same solution and consequently the search will be more diversified.

3.2 Global Updating Stage

When all ants have completed their solution, the pheromone level is updated by applying the global updating rule only on the paths that belong to the best solution since the beginning of the trail as follows:

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \Delta\tau_{ij} \tag{3}$$

where $\Delta\tau_{ij} = \begin{cases} \rho L_{gb} \text{ if } & (i,j) \in \text{best solution} \\ & \\ 0 & \text{otherwise} \end{cases},$

L_{gb} is the cost of the best solution from the beginning. This global updating rule is intended to provide a greater amount of pheromone on the paths of the best solution, thus intensifying the search around this solution. The transition probability to select the next item is given as:

$$p_{ij}^{k}(t) = \begin{cases} \frac{\tau_{ij}\eta_{ij}}{\sum_{j \in allowed_k(t)} \tau_{ij}\eta_{ij}} & \text{if } \ j \in allowed_k(t) \\ & \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

where τ_{ij} is a pheromone level to go from i to j, η_{ij} is the heuristic and $allowed_k(t)$ is the set of remaining feasible items. Thus the higher the value of τ_{ij} and η_{ij}, the more profitable it is to include item j in the partial solution.

Let $s_j = \sum_{i=1}^{m} r_{ij}$. For heuristic information we use:

$$\eta_{ij} = \begin{cases} p_j^{d_1}/s_j^{d_2} & \text{if } \ s_j \neq 0 \\ & \\ p_j^{d_1} & \text{if } \ s_j = 0 \end{cases} \tag{5}$$

Hence the objects with greater profit and less average expenses will be more desirable.

There are two possibilities for placing a pheromone. It can be placed on the arcs of the graph of the problem or on the nodes like in [7]. In the case of arcs τ_{ij}, in the formulas, corresponds to the pheromone quantity on the arc (i,j). In the case of nodes τ_{ij} corresponds to the pheromone quantity on the node j.

The LS method (move-by-move method) [10] perturbs a given solution to generate different neighborhoods using a move generation mechanism. In general, neighborhoods for large-sized problems can be much difficult to search. Therefore, LS attempts to improve an initial solution by a series of local improving changes. A move-generation is a transition from a solution S to another one $S' \in V(S)$ ($V(S)$ is a set of neighbor solutions) in one step. These solutions are selected and accepted according to some pre-defined criteria. The returned solution S' may not be optimal, but it is the best solution in its local neighborhood $V(S)$. A local optimal solution is a solution with the local maximal possible cost value. Knowledge of a solution space is the essential key to more efficient search strategies [11]. These strategies are designed to use this prior knowledge and to overcome the complexity of an exhaustive search by organizing searches across the alternatives offered by a particular representation of the solution. The main purpose of LS implementation is to speed up and improve the solutions constructed by the ACO.

A practical LS strategy that satisfies the MKP requirements has been developed and coupled with ACO algorithm. The MKP solution can be represented by string with 0 for objects that are not chosen and 1 for chosen objects. Our LS procedure is based on the exchange of the solution's components. This is a transition from one solution to another in one step. We use two exchange and the new solution is accepted if it is better than current solution.

4 Experimental Results

To explain the difference between the two proposed pheromone models we consider a small problem instance with $n = 5$ objects and one constraint.

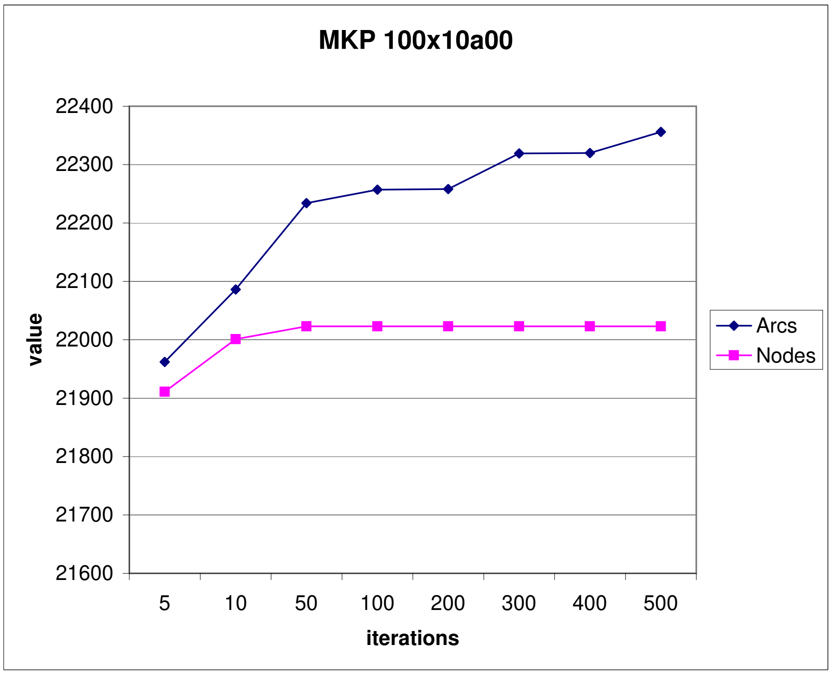
$$\begin{aligned} & max(5x_1 + 5x_2 + 4x_3 + 6x_4 + 6x_5) \\ & 5x_1 + 2x_2 + 3x_3 + 4x_4 + 3x_5 \leq 10 \end{aligned} \tag{6}$$

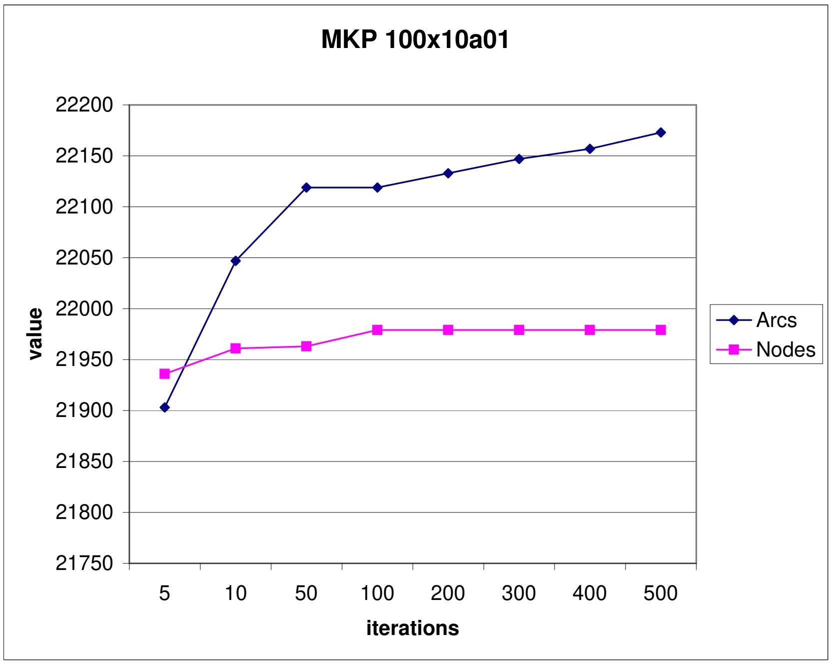
The optimal solution of this problem is (01011) and the optimal value of the object function is $L_{opt} = 17$. One ant will be used and it will start from node 1 of the graph of the problem in the first iteration and from nodes 3, 2 and 5 in the second, third and fourth iterations respectively, for both pheromone models. In the case of pheromone on the nodes, the achieved solutions are (11001) $L = 16$, (01101) $L = 15$, (11001) $L = 16$, (11001) $L = 16$. We observe that in iterations 3 and 4 the ant starts from nodes belonging to the optimal solution but it can not find the optimal one. In the case of pheromone on the arcs, the achieved solutions are (11001) $L = 16$, (01101) $L = 15$, (01011) $L = 17$, (01011) $L = 17$. In this case we observe that when the ant starts from the node belonging to the optimal solution it finds the optimal one.

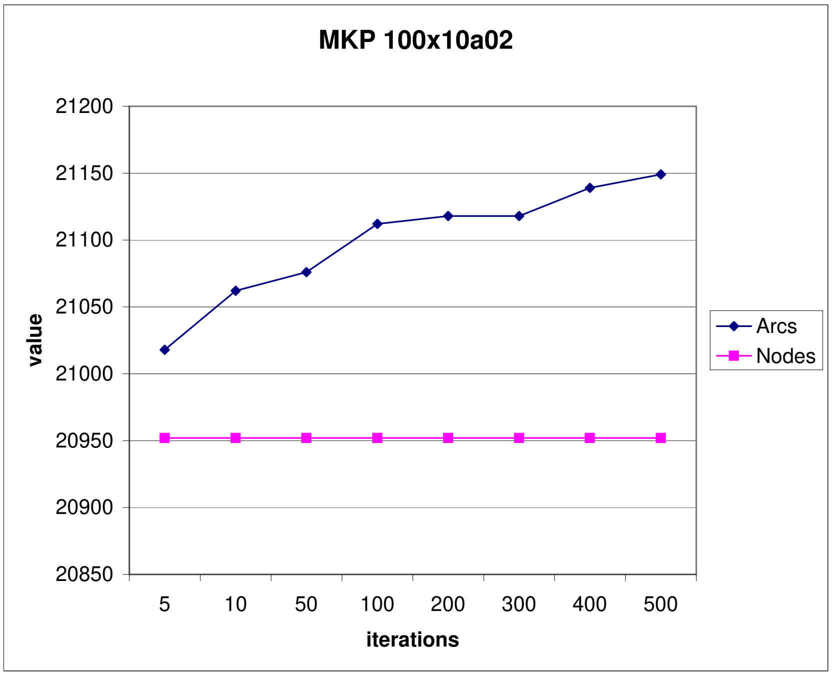
When the pheromone is on the nodes some of them accumulates large amount of pheromone and become much more desirable then all others. In the other case this pheromone is dispersed on the arcs that enter the node.

We show the computational experience of the ACS using MKP from "OR-Library" available within WWW access at *http://mscmga.ms.ic.ac.uk/jeb/orlib*. To provide a fair comparison for the above implemented ACS algorithm, a predefined number of iterations, K=500, is fixed for all the runs. The developed technique has been coded in $C++$ language and implemented on a Pentium 3 (450 MHz). Because of the random start of the ants we can use less ants then the number of the nodes (objects). After the tests we found that 10 ants are enough to achieve good results. Thus we decrease the running time of the program. The results on the Figure 1 shows the advantage of putting the pheromone on the arcs. The reported results are average value of the objective function over 20 runs of the program with different number of iterations. For all tested problems our algorithm achieves, in some runs, the best results found in the literature.

We observe that we have very early stagnation of the algorithm in the case of pheromone on nodes. We can explain this appearance with a large pheromone accumulation on some nodes and thus ants repeat same solution over and over


MKP 100x10a00
value
22400
22300
22200
22100
22000
21900
21800
21700
21600
5
10
50
100
200
300
400
500
iterations
Arcs
Nodes


MKP 100x10a01
value
22200
22150
22100
22050
22000
21950
21900
21850
21800
21750
5
10
50
100
200
300
400
500
iterations
Arcs
Nodes



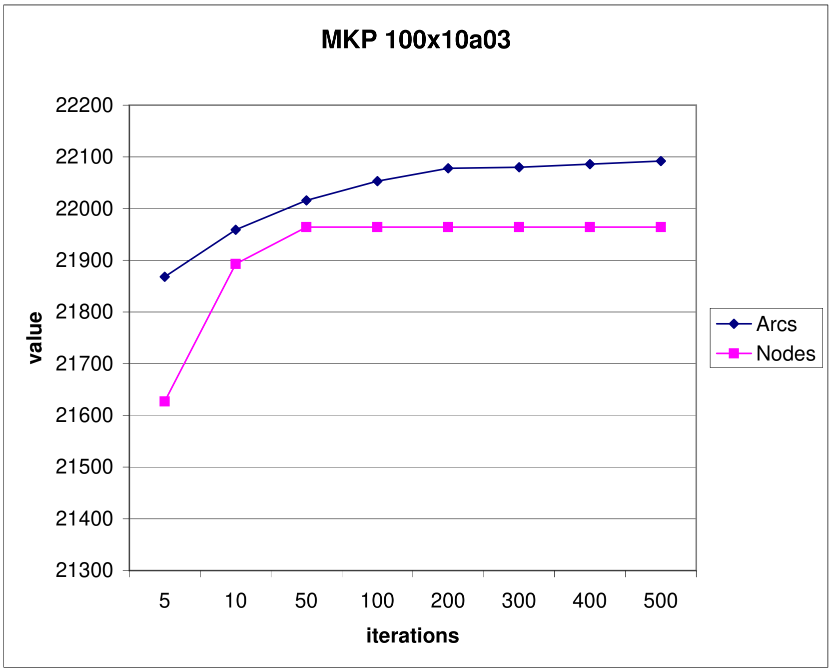


Fig. 1. The figures shows the average solution quality (value of the total cost of the objects in the knapsack) per iteration over 20 runs

again. In the case of pheromone on the arcs the difference between the amount of the pheromone on the arcs is not so big.

5 Conclusion

The design of a meta-heuristic is a difficult task and highly dependent on the structure of the optimized problem. In this paper two models of the pheromone placement have been proposed. The comparison of the performance of the ACS coupled with these pheromone models applied to different MKP problems are reported. The results obtained are encouraging and the ability of the developed models to rapidly generate high-quality solutions for MKP can be seen. For future work another important direction for current research is to try different strategies to explore the search space more effectively and provide good results.

Acknowledgments
Stefka Fidanova was supported by the European Community program "Center of Excellence"BIS 21 under contract number No ICA1-2000-70016 and the European Community program "Structuring the European Research Area" contract No MERG-CT-2005-510714..

References

1. Dorigo, M., Di Caro, G.:*The Ant Colony Optimization metaheuristic*.In: Corne, D., Dorigo, M., Glover, F.(eds.): New Idea in Optimization, McGrow-Hill (1999) 11–32.
2. Dorigo, M., Di Caro, G., Gambardella, L. M.:*Ant Algorithms for Distributed Discrete Optimization*. J of Artificial Life 5 (1999) 137–172.
3. Ferreira C. E., Martin A., Weismantel R.: *Solving Multiple Knapsack Problems by Cutting Planes*. SIAM Journal on Optimization Vol.6(3) (1996) 858–877.
4. Dorigo, M., Gambardella, L.M.:*Ant Colony System: A cooperative Learning Approach to the Traveling Salesman Problem*. IEEE Transaction on Evolutionary Computation 1 (1999) 53–66.
5. Gambardella, M. L., Taillard, E. D., Dorigo, M.: *Ant Colonies for the QAP*. J of Oper. Res. Soc. 50 (1999) 167–176.
6. Kochenberger, G., McCarl, G., Wymann, F.:*A Heuristic for General Integer Programming*. J of Decision Sciences 5 (1974) 34–44.
7. Leguizamon, G., Michalevich, Z.: *A New Version of Ant System for Subset Problems* In: Proceedings of Int. Conf. on Evolutionary Computations, Washington (1999).
8. Marchetti-Spaccamela A., Vercellis C.: *Stochastic on-line Knapsack Problems*J. of Mathematical Programming Vol 68(1) (1995) 73–104.
9. Osman, I. H., Kelley, J. P.:*Metaheuristic: An Overview*. In: Osman, I. H., Kelley, J.P. (eds.), Metaheuristic: Theory and Applications, Kluwer Academic Publishers (1996).
10. Schaffer, A. A., Yannakakis, M.: *Simple Local Search Problems that are Hard to Solve*. Society for Industrial Applied Mathematics Journal on Computing, Vol 20 (1991) 56–87.
11. Wolpert D., Macready W.: *No Free Lunch Theorems for Optimization*. In: IEEE Transactions on Evolutionary Computation,(1997) 67–82.