Probabilistic Model of Ant Colony Optimization for Multiple Knapsack Problem

Stefka Fidanova

Institute for Parallel Processing, Bulgarian Academy of Science, Acad. G. Bonchev 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 the probabilistic model in model-based search as ACO. We present the effect of four different probabilistic models for ACO algorithms 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 probabilistic 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 [2,4,3] is a meta-heuristic procedure for quickly and efficiently obtaining high quality solutions of complex optimization problems [11]. 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 [6]. 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

I. Lirkov, S. Margenov, and J. Waśniewski (Eds.): LSSC 2007, LNCS 4818, pp. 545–552, 2008.

MKP various probabilistic 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. In section 4 four probabilistic models are described. In Section 5 experimental results over test problems are shown. Finally some conclusions are drawn.

2 Formulation of the Problem

The Multiple Knapsack Problem has numerous applications in theory as well as in practice. It also arise as a subproblem in several algorithms for more complex COPs and these algorithms will benefit from any improvement in the field of MKP. We can mention the following major applications: problems in cargo loading, cutting stock, bin-packing, budget control and financial management may be formulated as MKP. In [12] there is proposed to use the MKP in fault tolerance problem and in [1] there is designed a public cryptography scheme whose security realize on the difficulty of solving the MKP. Martello and Toth [10] mention that two-processor scheduling problems may be solved as a MKP. Other applications are industrial management, naval, aerospace, computational complexity theory.

Most of theoretical applications either appear where a general problem is transformed to a MKP or where a MKP appears as a subproblem. We should mention that MKP appears as a subproblem when solving the generalized assignment problem, which again is used when solving vehicle routing problems. In addition, MKP can be seen as a general model for any kind of binary problems with positive coefficients [7].

The MKP can be thought as a resource allocation problem, where we have m resources (the knapsacks) and n objects. The object j has a profit p_j, each resource has its own budget c_i (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:

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

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

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

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 $I = \{1, \ldots, m\}$ and $J = \{1, \ldots, n\}$, with $c_i \ge 0$ for all $i \in I$. A well-stated MKP assumes that $p_j > 0$ and $r_{ij} \le c_i \le \sum_{j=1}^{n} r_{ij}$ for all $i \in I$ and $j \in J$. Note that the matrix $[r_{ij}]_{m \times n}$ and the vector $[c_i]_m$ are both non-negative.

In the MKP we are not interested in solutions giving a particular order. Therefore a partial solution is represented by $S = \{i_1, i_2, \ldots, i_j\}$ and the most recent elements incorporated to S, i_j need not to be involved in the process for selecting the next element. Moreover, solutions of ordering problems have a fixed length as we search for a permutation of a known number of elements. Solutions of 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. A fully connected graph means 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 clues) 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 above behavior of real ants has inspired ACO algorithm. This technique, which is a population-based approach, has been successfully applied to many NP-hard optimization problems [2,4]. The ACO algorithm uses a colony of artificial ants that behave as co-operative agents in a mathematical space where they are allowed to search and reinforce pathways (solutions) in order to find the optimal ones. A solution satisfying the constraints is said to be feasible.

```
procedure ACO
begin
  Initialize
  while stopping criterion not satisfied do
    Position each ant in a starting node
    repeat
      for each ant do
        Chose next node by applying the state transition rate
        Apply step-by-step pheromone update
      end for
    until every ant has build a solution
    Update best solution
    Apply offline pheromone updating
  end while
end
```

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 the 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 the Ant Colony System (ACS) [4] approach.

In ACS the pheromone updating stage consists of local update stage and global update stage.

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 the search is intensified around this solution.

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

$$\eta_{ij} = \begin{cases} p_j^{d_1}/s_j^{d_2} & \text{if } s_j \neq 0 \\ p_j^{d_1} & \text{if } s_j = 0 \end{cases} \tag{4}$$

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

The MKP solution can be represented by string with 0 for objects that are not chosen and 1 for chosen objects. The new solution is accepted if it is better than current solution.

4 Transition Probability

In this section we describe four possibilities for transition probability model. For ant k, the probability p_{ij}^k of moving from a state i to a state j depends on the combination of two values:

- The attractiveness η_{ij} of the move as computed by some heuristic.
- The pheromone trail level of the move.

The pheromone τ_{ij} is associated with the arc between nodes i and j.

4.1 Proportional Transition Probability

The quantity of the pheromone on the arcs between two nodes is proportional to the experience of having the two nodes in the solution. Thus the node j is more desirable if the quantity of the pheromone on arc (i, j) is high. For ant k which moves from node i to node j the rule is:

$$p_{ij}^k(t) = \begin{cases} \frac{\tau_{ij}\eta_{ij}(S_k(t))}{\sum_{q \in allowed_k(t)} \tau_{iq}\eta_{iq}(S_k(t))} & \text{if } j \in allowed_k(t) \\ 0 & \text{otherwise} \end{cases} , \tag{5}$$

where $allowed_k$ is the set of remaining feasible states, $S_k(t)$ is the partial solution at step t from ant k.

4.2 Transition Probability with Sum

This probability takes into account how desirable in the past has been the node j, independently how many ants have reached it from the node i or from some other. Thus the node j is more desirable if the average quantity of the pheromone on the arcs which entry in the node j is high. In this case the transition probability becomes:

$$p_{ij}^k(t) = \begin{cases} \frac{\left(\sum_{i=1}^n \tau_{ij}\right)\eta_{ij}(S_k(t))}{\sum_{q \in allowed_k(t)} \left(\sum_{l=1}^n \tau_{lq}\right)\eta_{iq}(S_k(t))} & \text{if } j \in allowed_k(t) \\ 0 & \text{otherwise} \end{cases} . \tag{6}$$

4.3 Maximal Transition Probability

This probability is proportional to the maximal pheromone on the arcs which entry in the node j. Thus the node j can be more desirable independently of the quantity of the pheromone on the arc (i, j) if there is some other arc with high quantity of the pheromone which entry in the node j. In this case the transition probability is changed as follows:

$$p_{ij}^k(t) = \begin{cases} \frac{(\max_l \tau_{lj})\eta_{ij}(S_k(t))}{\sum_{q \in allowed_k(t)} (\max_l \tau_{lq})\eta_{iq}(S_k(t))} & \text{if } j \in allowed_k(t) \\ 0 & \text{otherwise} \end{cases} . \quad (7)$$

4.4 Minimal Transition Probability

This probability is proportional to the minimal pheromone on the arcs which entry in the node j. Thus the node j will be more desirable if the quantity of the pheromone on all arcs which entry in the node j is high. In this case the transition probability is as follows:

$$p_{ij}^k(t) = \begin{cases} \frac{(\min_l \tau_{lj})\eta_{ij}(S_k(t))}{\sum_{q \in allowed_k(t)} (\min_l \tau_{lq})\eta_{iq}(S_k(t))} & \text{if } j \in allowed_k(t) \\ 0 & \text{otherwise} \end{cases} . \quad (8)$$

5 Experimental Results

In this section we describe the experimental analysis on the performance of MKP as a function of the transition probability. We show the computational experience of the ACS using 10 MKP instances from "OR-Library" available at `http://people.brunel.ac.uk/~{}mastjjb/jeb/orlib`, with 100 objects and 10 constraints. To provide a fair comparison for the above implemented ACS algorithm, a predefined number of iterations, $k = 400$, is fixed for all the runs. The developed technique has been coded in C++ language and implemented on a Pentium 4 (2.8 GHz).

Because of the random start of the ants in every iteration, we can use fewer ants than the number of the nodes. After the tests we found that 10 ants are enough to achieve good results. Thus we decrease the running time of the program. We run the same instance using different transition probability models on the same random sequences for starting nodes and we find different results. Thus we are sure that the difference comes from the transition probability. For all 10 instances we were running experiments for a range of evaporation rates and the parameters d_1 and d_2 in order to find the best parameters for every instance. We fixed the initial pheromone value to be $\tau_0 = 0.5$. After choosing for every problem instance the best rate for the parameters we could compare the different transition probabilities. In Figure 1 we show the average results over all 10 problem instances and every instance is run 20 times with the same parameter settings.

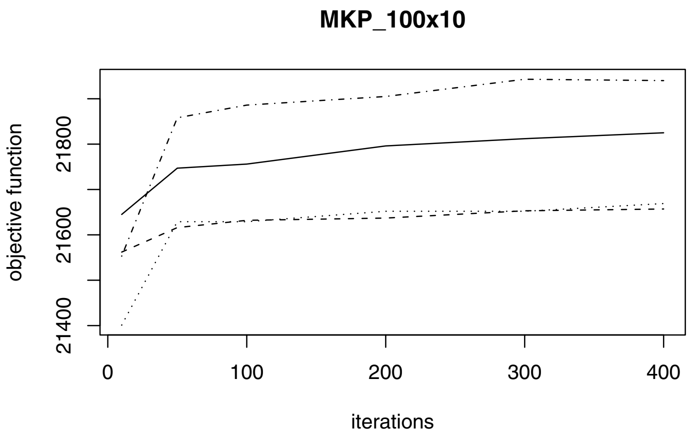

Fig. 1. The graphics shows the average solution quality (value of the total cost of the objects in the knapsack) over 20 runs. Dash dot line represents proportional probability, dash line — probability with sum, dots line — maximal probability and thick line — minimal probability.

Our first observation is that the proportional and the minimal transition probabilities show advantage over the sum and maximal transition probabilities. In a small number of iterations (less than 50), probability with sum and minimal probability achieve better results, but after that the proportional probability outperforms them. The MKP is not ordered problem. It means that the quality of the solution is not related to the order we choose the elements. Using maximal transition probability it is enough only one arc to have high quantity of the pheromone and the node will be more desirable. This kind of probability is more suitable to ordered problems: the node j is more desirable by node i than by node q. Using transition probability with sum the node is more desirable if the average quantity of the pheromone is high, but for some of the arcs this quantity can be very high and for other arc it can be very low and the average pheromone to be high. Thus we can explain the worst results with this two models of the transition probability. If the minimal probability is high, the quantity of pheromone for all arcs which entry the node is high. If the proportional probability is high it means that after node i is good to chose node j. The last two models of transition probability are better related to the unordered problems and thus we can achieve better results using them.

6 Conclusion

The design of a meta-heuristic is a difficult task and highly dependent on the structure of the optimized problem. In this paper four models of the transition probability have been proposed. The comparison of the performance of the ACS coupled with these probability models applied to different MKP problems are reported. The goal is to find probability model which is more relevant to the structure of the problem. The obtained results 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++.

References

1. Diffe, W., Hellman, M.E.: New direction in cryptography. IEEE Trans. Inf. Theory IT-36, 644–654 (1976)
2. Dorigo, M., Di Caro, G.: The Ant Colony Optimization metaheuristic. In: Corne, D., Dorigo, M., Glover, F. (eds.) New Idea in Optimization, pp. 11–32. McGrow-Hill (1999)
3. Dorigo, M., Di Caro, G., Gambardella, L.M.: Ant Algorithms for Distributed Discrete Optimization. J. of Artificial Life 5, 137–172 (1999)
4. Dorigo, M., Gambardella, L.M.: Ant Colony System: A cooperative Learning Approach to the Traveling Salesman Problem. IEEE Transaction on Evolutionary Computation 1, 53–66 (1999)
5. Ferreira, C.E., Martin, A., Weismantel, R.: Solving Multiple Knapsack Problems by Cutting Planes. SIAM Journal on Optimization 6(3), 858–877 (1996)
6. Gambardella, M.L., Taillard, E.D., Dorigo, M.: Ant Colonies for the QAP. J. of Oper. Res. Soc. 50, 167–176 (1999)
7. Kochenberger, G., McCarl, G., Wymann, F.: A Heuristic for General Integer Programming. J. of Decision Sciences 5, 34–44 (1974)
8. Leguizamon, G., Michalevich, Z.: A New Version of Ant System for Subset Problems. In: Proceedings of Int. Conf. on Evolutionary Computations, Washington, pp. 1459–1464 (1999)
9. Marchetti-Spaccamela, A., Vercellis, C.: Stochastic on-line Knapsack Problems. J. of Mathematical Programming 68(1), 73–104 (1995)
10. Martello, S., Toth, P.: A mixtures of dynamic programming and branch-and-bound for the subset-sum proble. Management Science 30, 756–771 (1984)
11. Osman, I.H., Kelley, J.P.: Metaheuristic: An Overview. In: Osman, I.H., Kelley, J.P. (eds.) Metaheuristic: Theory and Applications, pp. 1–21. Kluwer Academic Publishers, Dordrecht (1996)
12. Sinha, A., Zoltner, A.A.: The multiple-choice knapsack problem. J. of Operational Research 27, 503–515 (1979)