*BIO*AUTOMATION, 2009, **13** (4), 271-280

Ant Colony Optimization for Multiple Knapsack Problems with Controlled Starts

Fidanova S.[1*], Atanassov K.[2], Marinov P.[1], Parvathi R.[3]

[1] *Institute of Parallel Processing, Bulgarian Academy of Sciences*
25[A] Acad. G. Bonchev Str. 1113 Sofia, Bulgaria
E-mail: stefka@parallel.bas.bg
[2] *Centre of Biomedical Engineering, Bulgarian Academy of Sciences*
105 Acad. G. Bonchev Str. 1113 Sofia, Bulgaria
[3] *Department of Mathematics, Vellalar College for Women*
Erode – 638 012, Tamilnadu, India

Summary: Ant Colony Optimization is a stochastic search method that mimics the social behaviour of real ant colonies, which manage to establish the shortest routes to feeding sources and backwards. Such algorithms have been developed to reach near-optimum solutions of large-scale optimization problems, for which traditional mathematical techniques may fail. In this paper, a generalized net model of the process of ant colony optimization is constructed and on each iteration intuitionistic fuzzy estimations of the ants' start nodes are made. Several start strategies are developed and combined. This new technique is tested on Multiple Knapsack Problem, which is a real world problem. Benchmark comparisons among the strategies are presented in terms of quality of the results. Based on this comparison analysis, the performance of the algorithm is discussed along with some guidelines for determining the best strategy. The study presents ideas that should be beneficial to both practitioners and researchers involved in solving optimization problems.

Keywords: Ant Colony Optimization, Metaheuristics, Discrete optimization, Intuitionistic fuzzy estimation.

1. INTRODUCTION

The difficulties associated with using mathematical optimization on large-scale engineering problems have contributed to the development of alternative solutions. Linear programming and dynamic programming techniques, for example, often fail in solving NP-hard problems with large number of variables. To overcome these problems, researchers have proposed mataheuristic methods for searching near-optimal solutions to problems. One of the most successful metaheuristic is Ant Colony Optimization (ACO).

[*] *Corresponding author*

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 a real ant colony where the more the ants follow a trail, the more attractive that trail becomes.

ACO is inspired by real ant behavior to solve hard combinatorial optimization problems. Examples of hard optimization problems are Traveling Salesman Problem [3, 11], Vehicle Routing [12], Minimum Spanning Tree [9], Constrain Satisfaction [7, 11], Knapsack Problem [4], etc. The ACO algorithm uses a colony of artificial ants that behave as cooperative agents in a mathematical space where they are allowed to search and reinforce pathways (solutions) in order to find the optimal ones. The problem is represented by graph and the ants walk on the graph to construct solutions. The solutions are represented by paths in the graph. After the initialization of the pheromone trails, the ants construct feasible solutions, starting from random nodes, and then the pheromone trails are updated. At each step the ants compute a set of feasible moves and select the best one (according to some probabilistic rules) to continue the rest of the tour. The structure of the ACO algorithm is shown by the pseudocode below. The transition probability $p_{i,j}$, to choose the node j when the current node is i, is based on the heuristic information $\eta_{i,j}$ and the pheromone trail level $\tau_{i,j}$ of the move, where $i, j = 1, \ldots, n$.

$$p_{i,j} = \frac{\tau_{i,j}^{a} \eta_{i,j}^{b}}{\sum_{k \in Unused} \tau_{i,k}^{a} \eta_{i,k}^{b}}$$

where *Unused* is the set of unused nodes of the graph.

The higher the value of the pheromone and the heuristic information, the more probable 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; later, the ants update this value after completing the construction stage. ACO algorithms adopt different criteria to update the pheromone level.

```
Ant Colony Optimization
Initialize number of ants;
Initialize the ACO parameters;
while not end-condition do
    for k=0 to number of ants
        ant k chooses start node;
        while solution is not constructed do
            ant k selects higher probability node;
        end while
    end for
    Update-pheromone-trails;
end while
```

Fig. 1 Pseudocode for ACO

The pheromone trail update rule is given by:

$\tau_{i,j} \leftarrow \rho\tau_{i,j} + \Delta\tau_{i,j}$

where ρ models evaporation in the nature and $\Delta\tau_{i,j}$ is newly added pheromone, which is proportional to the quality of the solution.

The novelty in this work is the use of Intuitionistic Fuzzy Estimations (IFE, see [1]) of start nodes with respect to the quality of the solution and thus to better manage the search process. Various start strategies and their combinations are offered. The new technique is able to deal with real world problems. Like a benchmark problem is used Multiple Knapsack Problem (MKP) because a lot of real world problems can be represented by it and MKP arises as a subproblem in many optimization problems.

The rest of the paper is organized as follows: in section 2 several start strategies are proposed. In section 3 the strategies are applied on MKP and the achieved results are compared and strategies are classified. At the end some conclusions and directions for future work are proposed.

2. START STRATEGIES

The known ACO algorithms create a solution starting from random node. But for some problems, especially subset problems, it is important from which node the search process has started. For example, if an ant starts from node which does not belong to the optimal solution, the probability to construct it is zero. The aim is to

use the experience of the ants from previous iteration to choose the better starting node. Other authors use this experience only by the pheromone, when the ants construct the solutions. Therefore, the present paper offers several start strategies.

Let the graph of the problem has m nodes. The set of nodes is divided onto N subsets. There are different ways for dividing the set. Normally, the nodes of the graph are randomly enumerated. An example for creating the subsets, without lost of generality, is the following: the first number is in the first subset, the second node number– in the second subset, etc., the $N^{\text{-th}}$ node number is in the $N^{\text{-th}}$ subset, the $(N + 1)^{\text{-st}}$ node number is back in the first subset, etc. Thus there are almost equal number of nodes in the separate subsets. After the first iteration, the estimations $D_j(i)$ and $E_j(i)$ are introduced of the node subsets, where $i \geq 2$ is the number of the current iteration and $D_j(i)$ and $E_j(i)$ are weight coeffcients of the $j^{\text{-th}}$ node subset ($1 \leq j \leq N$), which are calculated by the following formulas:

$$D_j(i) = \frac{i.D_j(i-1) + F_j(i)}{i},$$

$$E_j(i) = \frac{i.E_j(i-1) + G_j(i)}{i},$$

where $i \geq 2$ is the current iteration and for each j ($1 \leq j \leq N$)

$$F_j(i) = \begin{cases} \dfrac{f_{j,A}}{n_j} & \text{if } n_j \neq 0 \\ F_j(i-1) & \text{otherwise} \end{cases}, \tag{1}$$

$$G_j(i) = \begin{cases} \dfrac{g_{j,B}}{n_j} & \text{if } n_j \neq 0 \\ G_j(i-1) & \text{otherwise} \end{cases}, \tag{2}$$

where $f_{j,A}$ is the number of the solutions among the best $A\%$, and $g_{j,B}$ is the number of the solutions among the worst $B\%$, where $A + B \leq 100$, $i \geq 2$ and $\sum_{j=1}^{N} n_j = n$, where n_j ($1 \leq j \leq N$) is the number of the solutions obtained by ants starting from nodes subset j. Initial values of the weight coefficients are: $D_j(1) = 1$ and $E_j(1) = 0$. Obviously, $F_j(i)$ and $G_j(i) \in [0; 1]$, i.e. they are IFEs.

Let threshold E for $E_j(i)$ and D for $D_j(i)$ be fixed; then the following five strategies to choose a start node for every ant are constructed, the threshold E increase at every iteration with $1/i$ where i is the number of the current iteration:

1. If $\frac{E_j(i)}{D_j(i)} > E$, then subset j is forbidden for **current iteration** and the starting node is randomly chosen from $\{j \mid j \text{ is not forbidden}\}$;
2. If $\frac{E_j(i)}{D_j(i)} > E$, then subset j is forbidden for **current simulation** and the starting node is randomly chosen from $\{j \mid j \text{ is not forbidden}\}$;
3. If $\frac{E_j(i)}{D_j(i)} > E$, then subset j is forbidden for K_1 **consecutive iterations** and the starting node is randomly chosen from $\{j \mid j \text{ is not forbidden}\}$ (where $0 \le K_1 \le$ "number of iterations" is a parameter);
4. Let $r_1 \in [R; 1)$ and $r_2 \in [0; 1]$ be random numbers. If $r_2 > r_1$, a node is randomly chosen from subset $\{j \mid D_j(i) > D\}$, otherwise a node is randomly chosen from the unforbidden subsets, R is chosen and fixed at the beginning.
5. Let $r_1 \in [R; 1)$ and $r_2 \in [0; 1]$ be random numbers. If $r_2 > r_1$, a node is randomly chosen from subset $\{j \mid D_j(i) > D\}$, otherwise a node is randomly chosen from the unforbidden subsets, R is chosen at the beginning and increases with r_3 at every iteration.

If $K_1 = 0$, then strategy 3 is identical to random choice of the start node. If $K_1 = 1$, then strategy 3 is equal to strategy 1. If K_1 = "maximal number of iterations", then strategy 3 is equal to strategy 2. Strategies 1, 2 and 3 can be called *forbidding strategies*, while strategies 4 and 5 can be called *stimulating strategies*. Under the stimulating strategies, the ants are forced to start their search from subsets with high value of $D_j(i)$. If $R = 0.5$, then the probability for an ant to start from nodes subset with high value of $D_j(i)$ is twice higher than the probability for it to start from another subset. For the forbidding strategies a fraction between $E_j(i)$ and $D_j(i)$ is used. This condition prevents some regions with several bad and several good solutions being forbidden.

More than one strategy for choosing the start node may be used, but there are strategies which cannot be combined. The strategies are distributed into two sets: St1 = {strategy 1; strategy 2; strategy 3} and St2 = {strategy 4; strategy 5}. The strategies from same set can not be used at once. Thus, a strategy from one of the sets can only be used or it can be combined with a strategy from the other set. Some sample combinations are (strategy 1), (strategy 2; strategy 5), (strategy 3; strategy 4).

3. EXPERIMENTAL RESULTS

This section analyzed the start strategy performance. The Multiple Knapsack Problem (MKP) is used for a test, because it is a real world subset problem and it has multiple applications in theory, as well as in practice. It also arises as a subproblem within several algorithms for more complex problems and these algorithms will further benefit from any improvement in the field of MKP. The following major applications can be mentioned as possible formulations of MKP: problems in cargo loading, cutting stock, bin-packing, budget control and financial management. Sinha and Zoltner in [10] proposed to use the MKP in fault tolerance problem and Diffe and Helman [2] designed a public cryptography scheme whose security relies on the difficulty of solving the MKP. Martello and Toth mention in [8] that two-processor scheduling problems may be solved as a MKP. Other applications are industrial management, naval, aerospace, computational complexity theory.

MKP can be thought as a resource allocation problem, where there are m resources (knapsacks) and n objects and every object j has a profit p_j. Each resource has its own budget c_j (knapsack capacity) and consumption $r_{i,j}$ of resource i by object j. The aim is maximizing the sum of the profits, while working within a limited budget.

MKP can be formulated as follows:

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

$$\text{subject to } \sum_{j=1}^{n} r_{i,j} x_j \le c_j, \quad i = 1, \ldots, m \tag{3}$$

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

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

With MKP the user is 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 be involved in the process for selecting the next element. Moreover, solutions for ordering problems have a fixed length as the user searches for a permutation of a known number of elements. Solutions of MKP, however, do not have a fixed length. The graph of the problem is defined as follows: the nodes correspond to the items, the arcs fully connect nodes. Fully connected graph means that after the object i the user can choose the object j, for every i and j if there are enough resources and object j has not been chosen yet.

The computational experience of the ACO algorithm is shown using ten MKP instances from the "OR-Library" available online at http://people.brunel.ac.uk/~mastjjb/jeb/orlib/, with 100 objects and 10 constraints. To provide a fair comparison for the above implemented ACO algorithm, a predefined number of iterations, $k = 100$, is fixed for all the runs. The developed technique has been coded in C++ language and implemented on a Pentium 4 (2.8 Ghz). The parameters are fixed as follows: $\rho = 0.5$, $a = 1$, $b = 1$, number of used ants is 20, $A = 30$, $B = 30$, $D = 1.5$, $E = 0.5$, $K_1 = 5$, $R = 0.5$, $r_3 = 0.01$. The values of ACO parameters (ρ, a, b) are from [5] and they are experimentally found to be the best for MKP. The tests are run with 1, 2, 4, 5 and 10 nodes within the nodes subsets. For every experiment, the results are obtained by performing 30 independent runs, then averaging the fitness values. The computational time for choosing start strategies is negligibly small compared to the computational time needed for obtaining the solutions.

Tests with all possible combinations of strategies and with random start (12 combinations) are run. Thus, the total number of tests is 18 000. One can observe that sometimes all nodes subsets become forbidden and the algorithm stops before performing all iterations (strategies 1, 2, 3 and combinations with them). When the nodes

subsets consist of 10 nodes, the algorithm does not perform all iterations for 80 of the strategies for 10 problems. In this situation there are two possibilities. The first possibility is to report the achieved solution when the algorithm stops. The second possibility is to continue the algorithm performance without any strategy, only applying a random start. The second possibility improves the achieved solutions with respect to the first one, so if all nodes subsets become forbidden the algorithm continues without a strategy.

For fair comparison of the achieved solutions by different strategies and node-devision, the difference d between the worst and best average result for every problem is divided by 10. If the average result for some strategy is between the worst average result and worst average plus $d/10$, it is evaluated with 1. If it is between the worst average plus $d/10$ and the worst average plus $2d/10$, it is evaluated with 2, and so forth. If it is between the best average minus $d/10$ and the best average, it is evaluated with 10. Thus, for a test problem the achieved results for every strategy and every nodes division is evaluated with a number from 1 to 10. Then, it is summed the rate of all test problems for every strategy and every nodes division. So the rate of the strategies/node-division becomes a number between 10 and 100, because the benchmark problems are 10 (see Table 1). It is a histogram like classification.

Table 1 shows that any strategy achieves better solutions for every node-division than random start. The highest rate is when the nodes subsets consist of 2 nodes, for most of the strategies. When the nodes subsets consist of 10 nodes the rate is low because from the same subset several very good and several very bad solutions can start. The highest rate (92) is obtained under strategy combinations 1-5 and 3-5 with two nodes in the nodes subsets. We can conclude that the best rate is produced by a combination of forbidding strategies and strategy 5, and it is better for the nodes subsets to be forbidden for a fixed number of iterations (1 or more), rather than for the whole simulation. The best obtained solutions are similar to the state-of-the-art solutions found in the literature in this field.

This modification of the ACO algorithm can be applied to other real world problems too. For example, ACO with start strategies for GPS surveying problem improves the achieved solutions with respect to the classical ACO with random start.

Table 1: Estimaton of strategies and nodes devision

number nodes	10	5	4	2	1
random	12	12	12	12	12
strategy 1	46	78	73	83	79
strategy 2	45	76	72	80	85
strategy 3	45	80	69	80	82
strategy 4	69	77	74	77	84
strategy 5	65	73	78	84	80
strategies 1-4	48	79	74	85	85
strategies 1-5	47	81	68	92	82
strategies 2-4	47	79	73	85	85
strategies 2-5	45	72	71	90	83
strategies 3-4	48	81	72	89	87
strategies 3-5	50	73	68	92	82

4. CONCLUSION

This paper is addressed on ant colony optimization algorithm with controlled start combining five start strategies. So, the start node of each ant depends on the goodness of the respective region. The achieved solutions with strategies are better than random start. Future work will be focused on parameter settings which manage the starting procedure. It will be investigated on influence of the parameters on the algorithm's performance. The aim is to study in detail the relationships between the start nodes and the quality of the achieved solutions.

ACKNOWLEDGEMENTS

This work has been supported by the National Science Fund of Bulgaria: Grant VU-MI-204/2006 (S. Fidanova, P. Marinov) and Grant BIn-2/2009 (K. Atanassov, R. Parvathi)

REFERENCES

1. Atanassov,K., Intuitionistic Fuzzy Sets, Springer, Heidelberg, 1999.
2. Diffe W., M. E. Hellman, New direction in cryptography. *IEEE Trans Inf. Theory*. IT-36, 1976, 644-654.

3. Dorigo M., L.M. Gambardella, Ant colony system: A cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation*, 1997, 1, 53-66.
4. Fidanova S., Evolutionary algorithm for multiple knapsack problem, Int. Conference Parallel Problems Solving from Nature, Real World Optimization Using Evolutionary Computing, ISBN No 0-9543481-0-9, Granada, Spain, 2002.
5. Fidanova S., Ant colony optimization and multiple knapsack problem, in: Renard, J.Ph. (Eds.), *Handbook of Research on Nature Inspired Computing for Economics ad Management*, Idea Grup Inc., ISBN 1-59140-984-5, 2006, 498-509.
6. Fidanova S., K. Atanassov, Generalized net model of the process of ant colony optimization, *Issues in Intuitionistic Fuzzy Sets and Generalized Nets*, 2008, 7, ISBN 978-83-88-311-91-8, 108-114.
7. Lessing L., I. Dumitrescu, T. Stutzle, A comparison between ACO algorithms for the set covering problem, ANTS Workshop, 2004, 1-12.
8. Martello S., P. Toth, A mixtures of dynamic programming and branch-andbound for the subset-sum problem, Management Science, 1984, 30, 756-771.
9. Reiman M., M. Laumanns, A hybrid ACO algorithm for the capacitate minimum spanning tree problem, *Workshop on Hybrid Metahuristics*, Valencia, Spain, 2004, 1-10.
10. Sinha A., A. A. Zoltner. The multiple-choice knapsack problem, *J. Operational Research* 1979, 27, 503-515.
11. Stutzle T., M. Dorigo, ACO algorithm for the traveling salesman problem, In Miettinen, K., Makela, M., Neittaanmaki, P., Periaux J. (Eds.), *EvolutionaryAlgorithms in Engineering and Computer Science*, Wiley, 1999, 163-183.
12. Zhang T., S. Wang, W. Tian, Y. Zhang. ACO-VRPTWRV: A new algorithm for the vehicle routing problems with time windows and re-used vehicles based on ant colony optimization, *Conference on Intelligent Systems Design and Applications*, IEEE press, 2006, 390-395.