

SOLVING MULTIPLE TRAVELLING SALESMAN PROBLEM USING MODIFIED ANT COLONY OPTIMIZATION ALGORITHM: A THEORITICAL APPROACH

Sushreeta Tripathy
Department of CS & IT, ITER,
Siksha'O'Anusandhan University,
Bhubaneswar- 751030, Odisha, INDIA

Abstract: Multiple travelling salesman problems (MTSP) is a typical complex combinatorial optimization problem, which is an extension of the travelling salesman problem (TSP). The MTSP can be generalized to a wide variety of routing and scheduling problems. It is known that classical optimization procedures are not adequate for this problem. The paper makes the attempt to show how the modified ant colony optimization (MACO) can be applied to the MTSP with ability constraint. For solving the multiple traveling salesman problem which utilizes an effective criterion for escaping from the local optimum points. The MTSP is one of the most important combinatorial optimization problems in which the objective is to minimize the distance traveled by several salesmen for servicing a set of nodes. Since this problem belongs to NP-hard Problems, some meta-heuristic approaches have been used to solve it. In contrast to the classical ACO, the proposed algorithm uses only a global updating for the current best solution and the best known found solution until now.

Keywords: Multiple Traveling Salesman Problem, Ant Colony Optimization, Updating Pheromone, Combinatorial Optimization Problems, Modified Ant Colony Optimization.

I. Introduction

Travelling Salesman Problem (TSP) is the most famous problem in the combinatorial optimization area [Lawler *et al* 1985]. And the multiple travelling problem (MTSP) is an extension of TSP. This problem relates to accommodating real world problems where there is a need to account for more than one salesman. The MTSP can be generalized to a wide variety of routing and scheduling problems, for example, the School Bus Routing Problem [Sayyah *et al* 2013] and the Pickup and Delivery Problem. Therefore, finding a good optimal solution method for the MTSP is important and induces to improve the solution of any other complex routing problems. However, MTSP is a NP-complete problem for which optimal solutions can only be found for small size problems. It is known that classical optimization procedures are not adequate for this problem. Good heuristic techniques are necessary for solving MTSP due to its high computational complexity. Modern heuristic techniques, namely genetic algorithm and ant colony optimization, can be good candidates for this problem. In this research, ant colony optimization (ACO) heuristic is used for solving ability limited MTSP. The main research contribution of this paper is to show how the MACO algorithm can be applied to MTSP with the limited ability, and the results are encouraging. In the following sections of this paper, the algorithm based ACO for MTSP is explained.

Ant colony optimization (ACO) [Dorigo *et al* 1996] is one of the most popular algorithms in the research field of swarm intelligence. ACO has been inspired by the behavior of real ants seeking a path between their colony and a source of food (fig. 1) [Majid *et al* 2013].

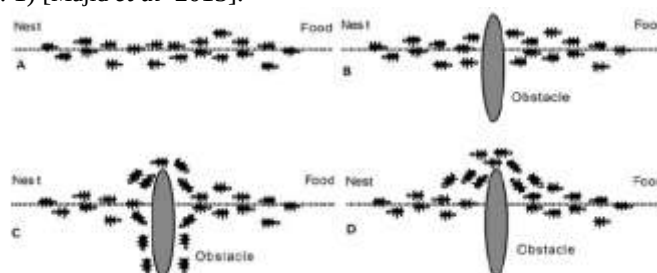


Fig. 1: (A) Real ants follow a path between the nest and a food source; (B) An obstacle appears on the path: Ants choose whether to turn left or right with equal probability; (C) Pheromone is deposited more quickly on the shorter path; (D) All ants have chosen the shorter path.

The ACO is an iterative stochastic search technique which was proposed by Marco Dorigo in 1992 . While walking between their colony and the food source, ants deposit pheromones along the paths they move. The pheromone level on the paths increases with the number of ants passing through and decreases with the evaporation of pheromone. As time passes, shorter paths attract more pheromone. Consequently, pheromone intensity helps ants to identify shorter paths to the food source. The first version of ACO called Ant System (AS) aimed at searching for an optimal path between two nodes in a graph. It should noted that a problem like traveling salesman problem (TSP) is divided into some sub-problems in which the simulated ants are expected to select the next node based on the amount of the pheromone in a trail and the distance to the next node. The probability of movement from node i to node j which is not visited yet by ant 'k' is presented by formula (1)[Majid *et al* 2013]

$$p_{ij}^k = \begin{cases} \frac{\tau_{ij}^\alpha \eta_{ij}^\beta}{\sum_{j \in N_i} \tau_{ij}^\alpha \eta_{ij}^\beta} & \text{if } j \in N_i^k \\ 0 & \text{if } j \notin N_i^k \end{cases} \quad (1)$$

Where:

N_i^k : The collection of nodes that hasn't been visited by ant 'k' yet

τ_{ij} : The value of pheromone on the arc joining i to j

η_{ij} The inverse distance between node i and j . However, both are powered by α and β which can be changed by the user. Therefore, their relative importance can be altered; α, β : The controlling parameters which are set by the user and determine the ratio of the importance of ant's visibility measure compared to the value of pheromone release on arc (i, j) .

Ants release Δ_{ij} called "pheromone information" on the respective path while moving from node i to j . This value can be calculated by formula (2).

$$\tau_{ij}(t) \leftarrow \tau_{ij}(t) + \Delta \tau_{ij} \quad (2)$$

If τ is the matrix for the existing pheromone on the edges of the respective graph, the algorithm like its natural version makes use of pheromone evaporation $1 - \rho$ in order to prevent the rapid convergence of ants to a sub-optimal path by formula (3). In other words, pheromone density on each edge is reduced in each iteration by $0 \leq \rho \leq 1$ which is set by the user.

$$\tau \leftarrow (1 - \rho)\tau \quad \rho \in [0, 1] \quad (3)$$

It's a good solution for small scale problems of TSP by AS (Fig. 2), this algorithm could not produce satisfactory results in large scale problems of TSP compared with meta-heuristic algorithms. Therefore, researchers attempted to develop a newer version of the algorithm which was able to produce better results. This algorithm has attracted the attention of an increasing number of researchers and several versions of ACO algorithms have been proposed in the literature such as , ant colony system (ACS) [Majid *et al* 2013], max-min AS (MMAS) [Stutzle and Hoos,2000], rank-based AS (RAS) [Bullnheimer *et al* 1997]. These algorithms are different in some ways, but all of them are based on a stronger exploitation of the search history to direct the ants' search process and share the same main idea which is the indirect communication between the individuals from a colony of ants, based on an analogy with the trails of pheromone.

II. Multiple Travelling Salesman Problem (MTSP)

The TSP is a well-known optimization problem in operations research and is great importance in other fields due to its widespread application in other problems. In this problem a salesman starts to move from a arbitrary node called node depot and returns after visiting n nodes so that each node is visited exactly once. The main objective is to find the shortest minimum cycle. One of the most important reasons for this attention is the fact that the TSP belongs to non-deterministic polynomial (NP-hard) problems in nature. It means that it requires a non polynomial time complexity at runtime to produce a solution. When the size of such problems increases, it requires an impractically enormous time is required to solve them. For this reason, numerous approaches have

been proposed to produce near-optimal solutions for TSP in an acceptable CPU time. Here we can found near optimal solution.

In this paper, improve on ACO according to the characteristic of the MTSP. One salesman should firstly travel an amount of cities, and the next salesman travels an amount of unvisited cities, in this way, all the salesmen travel the total cities. The travelled number of salesman is generated randomly in a certain range.

The multiple traveling salesman problem (MTSP) is a generalization of the well known TSP, where more than one salesman can be used in the solution. This problem is defined as follows. Let $G = (V, E)$ be a completely undirected connected graph with $v = \{0, 1, \dots, n\}$ as the set of vertexes and the set of arcs $E = \{(i, j) : 0 \leq i, j \leq n\}$ (if the graph is not complete, we can replace each missing arc with an arc which has an infinite size). Vertex 0 denotes the depot, and each vertex $i \in \{1 \dots n\}$ is a node. The cost of travel from vertex i to vertex j is denoted by C_{ij} ($C_{ij} = 0, 0 \leq i \leq n$ and it is assumed that costs are symmetric i.e. $C_{ij} = C_{ji}$). There are m salesmen at

the depot. Furthermore, each node must be serviced by a salesman and each salesman route must start at the depot and end in the depot. Here the main objective is to define the set of cycles which minimizes the total costs. An example is depicted in Fig. 2 [Pan and Wang, 2006] where the total numbers of sales man $m = 3$ and total number of cities $n = 7$. The MTSP can be stated as follows: There are m salesman who must visit a set of n cities, and each salesman is defined to start and end at the same depot. In this problem, each city must be visited exactly once by only one salesman and its objective is to find the minimum of total distances travelled by all the salesmen.

Fig. 2: Multiple Travelling Salesmen Problem

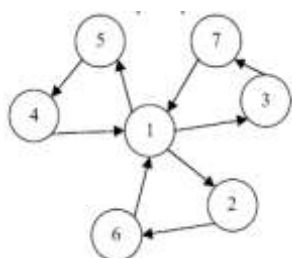
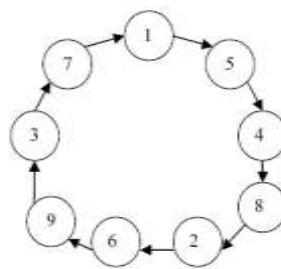


Fig. 3: Transformation from MTSP to TSP



The general objective of the MTSP is to minimize the total distance, there are $m-1$ cities always to select the nearest cities as their round trip. As a result, TSP which is made up of the left $n-m+1$ cities is left. During the m salesmen, there are $m-1$ salesmen travelling only one city, and one salesman needs to travel the left $n+m-1$ cities. This is not up to the mustard. In practice, every salesman has the similar ability, and the limit in ability. So the MTSP with ability constraint is more appropriate in the real world problems. In this paper, we suppose the number of cities which are travelled by every salesman is limited. There is a wide variety of MTSPs and an extensive literature on this class of problems. The variants include MTSP with pickup and delivery (If the salesmen need to pick up loads) [Wang and Regan, 2002], multi depot MTSP (if the problem has a single depot or multiple depots) [Rathinam and Sengupta, 2006] and others. The methods for solving MTSP can be classified as exact, heuristic, meta-heuristic and hybrid algorithms. Although there are exact algorithms which investigate fewer solutions and can obtain optimal solutions for small size problems, it is almost impossible even in such cases to find an optimum solution within a satisfactory time limit. There are several exact algorithms of the MTSP such as Cutting Planes algorithm [Laporte and Nobert, 1980], Branch-and-Bound algorithm and Lagrangian algorithm. For instance, many researchers have tried to relax the MTSP to the TSP and apply algorithms to resolve it, but the results have always ended in degeneration. In this method, the MTSP [Ai and Kennington 1986] with m salesmen and n cities is transformed into a TSP with $n + m - 1$ cities through introducing $m - 1$ artificial depots ($n + 1 \dots n + m - 1$).

The TSP belongs to the class of NP-hard problems and MTSP [Malik *et al* 2007] is more difficult than TSP because it involves finding a set of Hamilton circuits without sub-tours for $m(m > 1)$ salesmen who serve a set of $n(n > m)$ cities so that each one is visited by exactly one salesman. Consequently, it is obvious that MTSP also belongs to NP-hard problems. This means that the MTSP solution time grows exponentially with the increase in distribution points. As a result, that classical optimization procedures are not adequate for this problem especially for large dimensions. In other words, as the size of the problem increases, the complexity of the problem also increases rapidly as well. A lot of algorithms have been performed on the MTSP including heuristic approaches such as k -opt approach [Potvin *et al* 1989], minimum spanning tree [13], and self-organizing NN approach.

Although heuristic methods solve NP-hard problems, they become trapped in local optima and cannot obtain a high quality solution. Consequently, due to MTSP's high computational complexity, good meta-heuristic techniques namely genetic algorithms (GA), neural network (NN), tabu search (TS), simultaneous generalized

hill climbing algorithms (SGHC), and ACO are necessary to solve it.. These algorithms basically try to combine basic heuristic methods into higher level frameworks in order to explore a search space efficiently

III. The Proposed Modified Ant Colony Optimization Algorithm for MTSP

In this section the new modified ACO called MACO algorithm is described. There are several modifications in the MACO compared to ACO including transition rule, candidate list, global pheromone updating rules and several local search techniques. In contrast to local searches, using the candidate lists leads to the concentration of the search on promising nodes in order to reduce the computational time. These modifications help to avoid premature convergence and then to search over the subspace in order to find the global optimum. The steps of the proposed algorithm are mainly described as follows:

Step 1: Build n number of solution of the MTSP by using a new transition rule and a candidate list.

Step 2: Use insert, swap and 2-opt moves for the current best solution and the best solution until now in order to improve them more.

Step 3: Update the global pheromone information.

At the first stage of the MACO, n ants are initially positioned on the depot and each ant builds a tour for each salesman by using a candidate list and formula (4). This step is iterated for n times. Therefore, n feasible solutions for the MTSP are partially obtained in this step. The MACO differs from the ACO due to its strategy of new transition rule. In the proposed algorithm, the next node j from node i in the route is selected by ant k , among the unvisited candidate nodes O_i^k with n number, according to the following transition rule which shows the probability of each city being visited. In other words, the ant k at point i maintains a tabu list N_i^k in memory which defines the set of points still to be visited when it is at this point.

$$p_{ij}^k = \begin{cases} \frac{\tau_{ij}^\alpha \eta_{ij}^\beta k_{ij}^\lambda}{\sum_{j \in N_i^k} \tau_{ij}^\alpha \eta_{ij}^\beta k_{ij}^\lambda} & , \text{ if } j \in O_i^k \\ 0 & , \text{ if } j \notin O_i^k \end{cases} \quad (4)$$

Where $\tau_{ij}(t), \eta_{ij}(t)$, α and β are defined in formula (1) as well. Moreover, k_{ij} is defined as the savings of combining two nodes on one tour as opposed to serving them on two different tours and λ is a control parameter set by the user. The savings of combining any two nodes i and j are computed by

$$k_{ij} = \alpha c_{i0} + \beta c_{0j} - \lambda c_{ij} \quad \text{where node } 0 \text{ is the depot and } c_{ij} \text{ denotes the distance between nodes } i \text{ and } j$$

.A candidate list is used to improve the quality of the solutions and computational time in the optimization problems. For example, it is used to determine the next customer which is not visited until now and is selected in a vehicle route based on the distance to all other customers in the customer set. In the proposed algorithm, the closest unvisited nodes are only considered in the candidate list for the current node i and are made available for selection as the next node to be visited in the route. It should be noted that the size of the candidate list has been determined by restricting its size to a fraction of the total number of nodes in the problem. It is noted that this restriction prevents the algorithm from wasting its efforts to consider moves to nodes which are a great distance from the current node and have very little chance of creating an improved solution to the problem. For some optimization problems, ACO even can obtain the best solutions and outperform other meta-heuristics. All these powerful ACO algorithms have been applied in these problems with local search methods. This is usually achieved by improving the best solution through local search. In the form of effectiveness and efficiency, three simple procedures including insert, swap and 2-opt moves are adopted in this study. After all ants have constructed their routes and before updating global pheromone, several local searches are performed in an attempt to increase the length of the routes. Local searches are conducted in single and multiple route. In single insert and single swap moves, the nodes and destination positions are selected in a way that operation causes the greatest improvement in the cost of the solution. In a multiple insert, a candidate node i is removed from its current route (origin) and a trial insertion is made in the other route if the destination route contains at least one of the δ -nearest neighbors of i (δ is defined by the user). Furthermore, in multiple swap move, a candidate node i of $R1$ is exchanged with j of $R2$ if one of the destination routes contains at least one of the δ -nearest neighbor to the acceptable selected node.

The 2-opt move is implemented in one route or is applied to two routes. This algorithm is one of the most commonly encountered moves in which two edges belong to the same route or different routes which form a

criss-cross are selected and two new edges are replaced. It also should be noted that the new solution will be only accepted when constraints of the MTSP are not violated and the novel tour will gain better value for problem than the previous solution.

The basic step of 2-opt is to delete two edges from a tour and reconnect the remaining fragments of the tour by adding two new edges. See the figure below for an example. Once we choose the two edges to delete, we do not have a choice about which edges to add – there is only one way to add new edges that results in a valid tour.

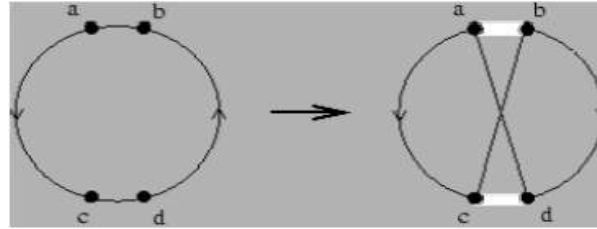


Fig. 4: Example of 2-opt

The 2-opt algorithm repeatedly looks for 2-opt moves that decrease the cost of the tour. A 2-opt move is the same as inverting a subsequence of cities in the tour. For example, in the picture above, the initial tour is $a, \dots, c, d, \dots, b, a$ while the final tour is $a, \dots, c, b, \dots, d, a$. So the segment $d, \dots, b$ is inverted in the modified tour.

A. Pseudocode for the 2-opt local search algorithm:

current_tour := create_random_initial_tour()

repeat

modified_tour := apply_2opt_move (current_tour)

if length(modified_tour) < length(current_tour)

then current_tour := modified_tour until no further improvement or a specified number of iterations

Although the 2-opt algorithm performs well and can be applied to Multiple Traveling Salesperson problems with many cities. The local searches stop until no additional improvement can be obtained. Furthermore, a local search is a time-consuming procedure of ACO. To save the computation time, we will only apply a local search to the current best solution and the best known solution until now. The idea here is that better solutions may have a better chance of finding a global optimum.

The pheromone updating formula was used to simulate the change in the amount of pheromone due to both the addition of a new pheromone deposited by ants on the visited edges and the pheromone evaporation. Using this rule, ants will search in a wide neighborhood of the best previous schedule. Another problem with all of the versions of the ACO is the fact that pheromone is deposited only on the best current route or only on the best known solution in global pheromone release. Due to the random nature of ACO, the value of the best solution function might decrease or increase in different iterations. Therefore, it is likely that in a few iterations the value of objective function does not increase and despite the fact that a solution is not good enough (especially at the start of algorithm) it is deposited with pheromone repeatedly in several iterations. Consequently, in following iterations this additional pheromone on an inappropriate solution will cause the algorithm to waste a lot of time on changing the search direction in solution space without being able to find a better solution. As a result, the pheromone of all edges belonging to the route taken by ants will be updated in the proposed MACO. Besides, the pheromone is increased only on edges which belong to two best found solutions in current iteration and since the beginning. The idea of the proposed strategy in the context of the proposed algorithm is to place extra emphasis on the two best paths found after each iteration. This modification leads to a balance between exploitation (through emphasizing the global best ant) as well as exploration (through the emphasis to iteration best ant).

As mentioned above, in the proposed algorithm, pheromone is released on the edges of the current best solution

T^{cbs} and the best known solution until now T^{bks} with coefficient e in each iteration based on the formula (5):

$$\Delta \tau_{ij}^{cbs(bks)}(t) = \begin{cases} \frac{e}{L^{cbs}(t)} \left(\frac{e}{L^{bks}(t)} \right) & , (i,j) \in T^{cbs(bks)} \\ 0 & , (i,j) \notin T^{cbs(bks)} \end{cases} \quad (5)$$

In this way, the edges of the shortest path up to the current iteration including T^{cbs} and T^{bks} become more attractive and are updated based on the values of the best L^{cbs} and L^{bks} tours. It should be noted that the lower the value of the tours in formula (6), the greater the pheromone deposited on the visited edges. The updating formula of the pheromone in the proposed MACO algorithm is:

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \Delta\tau_{ij}^{cbs}(t) + \Delta\tau_{ij}^{bks}(t) \quad (6)$$

where the parameter $0 < \rho < 1$ is the persistence of the pheromone trail so that $1-\rho$ indicates the evaporation rate and is the amount of pheromone which an ant deposits on the trail. The pheromone updating reflects the desirable factors on that trail such as shorter distance, better performance, etc., depending on the application problem. Furthermore, it results in the new pheromone trail being a weighted average between the old pheromone value and the amount of pheromone deposited. It should be noted that if the best solution till now does not improve within a given ten steps in the proposed algorithm, the algorithm will be stopped. Steps of MACO for the MTSP is presented below:

Steps:

- 1) Set all the parameters.
- 2) Place all ants at the depot.
- 3) Select a nest node for each ant based on a new proposed formula and efficient candidate list.
- 4) Deposit pheromone by each ant when they move.
- 5) If there is a node that has not been visited till now, go to step- 3.
- 6) Save the current best solution and its value obtained in current iteration.
- 7) Update the current best solution and its value obtained until now.
- 8) Apply the local searches for two known best solutions in current iteration and obtained until now.
- 9) Update global pheromone on each edge.
- 10) If the best solution till now is improved within ten iterations, go to step 2.
- 11) Print the best solution and its value.

IV. Conclusion

In this paper, a new modified version of ACO called MACO which employs several effective modifications in order to solve the MTSP was presented. The modifications improved the performance of the classic ACO in escaping from local optimum points and finding better solutions in comparison with the other metaheuristic algorithms. Furthermore, the proposed algorithm can be used for other combinatorial optimization problems like vehicle routing problem and the capacitated clustering problem. MACO is a promising optimization technique for solving complex combinatorial optimization problems like the MTSP. In the past, ACO has been applied to several combinatorial optimization problems successfully.

References

- [1] Ai A.; Kennington, J.L.(1986). *Exact solution of multiple traveling salesman problems*, Discrete Applied Mathematics, 13, 1986, pp. 259–276.
- [2] Bullnheimer, B.; Hartl, R.F.; Strauss, C.(1997). *A new rank based version of the ant system – A computational study*, Central European Journal for Operations Research and Economics, 7(1), pp. 25–38, 1997.
- [3] Dorigo, M.; Maniezzo, V.; Colomi A.(1996). *The ant system: optimization by a colony of cooperating agents*, IEEE Transactions on Systems, Man, and Cybernetics–Part B, 26(1), pp. 29–41.
- [4] Lawler, E.L.; Lenstra, J.K.; Kan, A.H.G.; Shmoys, D.B.(1985). *The traveling salesman problem: a guided tour of combinatorial optimization*, Chichester, Wiley, 11.201-209.
- [5] Laporte, G.; Nobert, Y.A.(1980). *Cutting planes algorithm for the m-salesmen problem*, Journal of the Operational Research Society, 31, pp. 1017–1023.
- [6] Malik, W.; Rathinam, S.; Darbha, S.(2007). *An approximation algorithm for a symmetric Generalized Multiple Depot, Multiple Travelling Salesman Problem*, Operations Research Letters, 35(6), pp. 747–753.
- [7] Majid, Y.; Farzad, D.; Farhad, R.(2013). *Modification of the Ant Colony Optimization for Solving the Multiple Traveling Salesman Problem*, Romanian Journal of Information Science and Technology, Volume 16, Number 1, pp 65–80.
- [8] Potvin, J.; Lapalme, G.; Rousseau, J.(1989). *A generalized k-opt exchange procedure for the MTSP*, Information Systems and Operations Research, 27(4), pp. 474–481.
- [9] Pan, J.; Wang, D.(2006). *An Ant Colony Optimization Algorithm for Multiple Travelling Salesman Problem*, First International Conference on Innovative Computing, Information and Control (ICIC'06), IEEE.
- [10] Rathinam, S.; Sengupta, R.(2006) *Lower and Upper Bounds for a Symmetric Multiple Depots, Multiple Travelling Salesman Problem*, Research Report, UCB-ITS-RR-2006-2.
- [11] Sayyah, M.; Larki, H.; Yousefikhoshbakht, M.(2013). *Solving the Vehicle Routing Problem with Simultaneous Pickup and Delivery by an Effective Ant Colony Optimization*, Transportation Research Journal, 1, pp. 1–12.
- [12] Stutzle, T.; Hoos, H.H.(2000). *MAX-MIN ant system*, Future Generation Computer System, pp. 889–914.
- [13] Wang, X.; Regan, A.C.(2002). *Local truckload pickup and delivery with hard time window constraints* Transportation Research Part B, 36, pp. 97-112.