Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 5 Issue 02, February-2016

Solving Multiple TSP Problem by K-Means and Crossover based Modified ACO Algorithm

Majd Latah
Graduate Student,
Department of Computer Engineering
Ege University
Izmir – Turkey

***Abstract*— The travelling salesman problem (TSP) is a famous combinatorial optimization problem where a salesman must find the shortest route to n cities and return to a home base city. While the TSP restricts itself to one salesman, the mTSP generalizes the problem to account for more than one salesman. A natural approach for solving this kind of problems is to group the cities in clusters where each cluster represent a set of adjacent cities then to use one of the well know optimization approaches for finding the optimal path route for each cluster we have. In this paper we introduce a modified Ant colony optimization algorithm for solving the mTSP problem. We used the standard TSPLIB data set to compare the performance of the proposed algorithm with both basic ACO and genetic algorithm approaches and our algorithm showed effective results compared by the basic ACO algorithm and competitive results to the genetic algorithm approach.**

***Keywords*— mTSP, K-means, Ant colony optimization, ACO.**

I. INTRODUCTION

The Travelling Salesman Problem (TSP) is a well known and important combinatorial optimization problem. The solution of this problem is to find the shortest path that visits each city in a given list exactly once and then returns to the starting city. The multiple traveling salesman problem expands the traditional TSP to allow for multiple salesmen. Thus, each city must be visited exactly once by any salesman. In this approach, cities are clustered together and assigned to different salesman, thus converting the Multiple TSP problem into n simple TSP problem. Multiple TSP has many application areas such as the routing problems, the Pickup, Delivery Problem (Christofides [1], Savelsbergh [2]), print press scheduling [3] and crew scheduling [4]. Therefore, finding an efficient algorithm for the mTSP problem is important and induces to improve the solution of any other complex routing problems. Also the mTSP can be used to solve the problem of multiple traveling robots [5,6]. In mTSP problem we have an undirected connected graph including E nodes connecting between V vertices; every salesman will visit a subset of nodes and return to returns to single depot vertex which will produce a tour similar to the simple form of TSP problem. The total cost for visiting the nodes by each salesman man must be minimized. For each salesman we have to assign the optimal ordering of the nodes that represents the salesman's tour. The TSP belongs to the class of NP-hard problems [7] and MTSP 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 will be visited by exactly one salesman.[8]

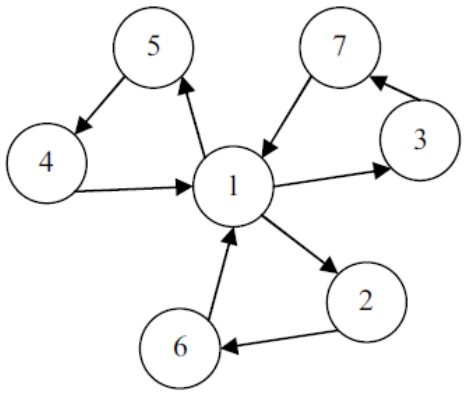

Fig 1. A solution of simple mTSP [15]

II. RELATED WORKS

In last few decades, many approaches have proposed to solve the mTSP problem. Like the solution in [17] is based on branch and bound algorithm. However, because of the combinatorial complexity of mTSP problem there is a need to apply heuristic methods especially for large scale instances of mTSP. In [18] the first heuristic approach was applied. Another neural network based approach was presented in [19]. In [22] also a Tabu Search (TS) algorithm which is a metaheuristic approach used for solving mTSP. Later, genetic algorithms were also used for solving the mTSP problem [20][21].

III. K-MEANS ALGORITHM

One of the most popular clustering methods is k-means clustering algorithm. In K means n observations will be partitioned into k clusters in a way that each observation belongs to the cluster with the nearest mean. First k initialized based on the desired number of clusters. Each data point is assigned to nearest centroid and the set of points assigned to the centroid is called a cluster. Each cluster centroid is updated based on the points assigned to the cluster. The process will be repeated until the centroids remain the same or no point changes clusters. In this algorithm mostly Euclidean distance is used to find distance between data points and centroids. The main drawback of K-means algorithm is the quality of the clustering results highly depends on random selection of the initial centroids. For different runs it gives different clusters for the same input data.

Formally, the goal is to partition the n entities into k sets Si, i=1, 2, ..., k in order to minimize the within-cluster sum of squares (WCSS), defined as:

$$\sum_{j=1}^{k}\sum_{i=1}^{n} \| x_i^j - C_j \|^2 \qquad (3.1)$$

where term $|x_i^j - C_j|$ provides the distance between an entity point and the cluster's centroid. The K-Means is a greedy, computationally efficient technique, being the most popular representative-based clustering algorithm. The k-means clustering algorithm is as follows:

1. Initialize cluster centroids $\mu_1, \mu_1, \ldots\ldots, \mu_k \in R^n$ randomly.

2. Repeat until convergence: {

For every i, set

$$C^{(i)} := \arg\min_j \|x^{(j)} - C_j\|^2$$

For each j, set

$$\mu_j = \frac{\sum_1^m 1\{C^{(i)} = j\} x^{(i)}}{\sum_{n=1}^m 1\{C^{(i)} = j\}} \quad \}$$

IV. Ant Colony Optimization

Ant Colony Optimization (ACO) is a swarm intelligence method that uses metaheuristic optimizations. The main principle of ACO is the natural behaviour of ants when they seek a path between their colony and a source of food. When one of the tasks ants seeking for food it deposits a chemical substance called pheromone in the ground. This substance helps the ant to find the return way and allow other ants to know the way they have taken. The main idea is that pheromone density becomes higher on shorter paths. Pheromone evaporation helps also in avoiding the convergence to a locally optimal solution. Because pheromone represents a positive feedback so when one ant finds a good path the other ant will more likely follow it and also give a feedback about the quality of that path. [10]

The first ACO algorithm was called Ant System (AS) [11]. Later many developments appeared like the Ant Colony System (ACS) by Dorigo and Gambardella [12]. The pseudo-code for the ACO algorithm is as follows:

The Ant Colony Optimization Metaheuristic:

```
    Initialize parameters,
    initialize pheromone trails
    while termination condition not met do
    ConstructAntSolutions
    ApplyLocalSearch (optional)
    UpdatePheromones
    end while
```

The difference between AS and ACS is the transition rule, In AS we use the transition rule which is given in (3.1).

$$p_{ij}(t) = \begin{cases} \dfrac{[\tau_{ij}(t)]^\alpha [\eta_{ij}]^\beta}{\sum_{h\in\Omega}[\tau_{ih}(t)]^\alpha [\eta_{ih}]^\beta} & if \quad j \in \Omega \\ 0 & otherwise \end{cases} \qquad (4.1)$$

Where:

τ_{ij} is the amount of pheromone trail between on edge *i,j,*

η_{ij} is the desirability or visibility of edge i,*j* ($\eta_{ij} = 1 / d_{ij}$),

α is a parameter to control the influence of τ_{ij},

β is a parameter to control the influence of η_{ij} ,

d_{ij} is the distance between city *i* and city *j,*

Ω is the set of unvisited cities.

Whereas in ACS we use another transition rule which is given below:

$$\Delta\tau_{ij}^k(t) = \begin{cases} [\tau_{ij}(t)][\eta_{ij}]^\beta & if\ q \le q_0 \\ S, & otherwise \end{cases} \qquad (4.2)$$

Where q is a random variable uniformly distributed in [0, 1]. According to (4.2) ant k at city i chooses to move to city j with best edge using pheromone trail and heuristic value with q0 probability. Otherwise it selects a random city in its neighbourhood using a transition rule as mentioned in (4.1). Also another difference we see in (4.2) is that the α parameter has been set as constant in the best value which is 1.[16]

When all the ants have completed a solution, the trails are updated by:

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \sum_{k=1}^{m} \Delta\tau_{ij}^k(t) \qquad (4.3)$$

Where τ_{ij} is the amount of pheromone deposited for a state transition ij and ρ is the pheromone evaporation coefficient. While $\Delta\tau_{ij}^k(t)$ is the amount of pheromone deposited by k_{th} ant typically given for a TSP problem by:

$$\Delta\tau_{ij}^k(t) = \begin{cases} Q / L^k(t) & if\ edge(i,j)\ is\ used\ by\ ant\ k \\ 0 & otherwise \end{cases} \qquad (4.4)$$

Where L_k is the cost of the k_{th} ant's tour (typically length) and Q is a constant. It is assumed that for every tour all ants have the same pheromone amount. In this way, the ants with the shortest tour store more pheromone.

While we run the ACO algorithm, the value of the pheromones may goes extremely small or high. If the value is very small that means the correspondence arc mostly will not be selected again, also if it is very large that will lead to construct the same solution again. That is why it has been suggested to set upper and lower bounds on the pheromones.

The authors in [23] defined the first mechanism for the pheromone bounds as shown below:

$$\tau_{\max} = \frac{1}{\rho F^*} \qquad (4.5)$$

Where F* is the total cost of the best solution found so far and ρ is the pheromone evaporation coefficient.

$$\tau_{\min} = \frac{\tau_{\max}(1-P_{dec})}{(\frac{n}{2}-1)P_{dec}} \tag{4.6}$$

Where P_{dec} is the probability of constructing the best solution and n is the number of the components of the solution. When a new global best solution is found, $\tau_{\min}$ must be updated. [24]

To improve the results of ACO we can apply local search or modify the number of ants. But it has a disadvantage which is the computational time.

V. The Proposed Algorithm

In this section, we propose an algorithm that combines the benefits of k-means clustering and ACO algorithm for solving the mTSP problem. In our algorithm in the first step we make m cluster using k-means algorithm where m is equals to the number of the salesmen in the mTSP problem and for each cluster we solve it independently using the modified ACO algorithm. We modified the ACO algorithm as follows:

1. The proposed pheromone update rule is:

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t)\frac{z}{10*(\lfloor\log_{10} z\rfloor+1)} + \sum_{k=1}^{m}\Delta\tau_{ij}^{k}(t) \tag{5.1}$$

$$z = \sum_{k=1}^{m} L_k \tag{5.2}$$

Where L_k is the cost of the k_{th} ant's tour. In this way, the decreasing of the pheromone will be less as we go near by more optimized solutions.

2. Applying a crossover operation pheromone update:

In genetic algorithms, crossover (also called recombination) combines parts of two or more parental solutions to create new, possibly better solutions. Crossover point(s) is determined stochastically. The crossover operation could be in a single point or double point as shown in the Figure 2 and Figure 3.

Parent 1	1	0	0		1	0	0	1	1	0
Parent 2	0	1	1		0	0	1	0	0	1

Child 1	1	0	0		0	0	1	0	0	1
Child 2	0	1	1		1	0	0	1	1	0

Fig 2. Single Point Crossover

Parent 1	1	0	0		1	0	0	1		1	1	0
Parent 2	0	1	1		0	0	1	0		0	0	1

Child 1	1	0	0		0	0	1	0		1	1	0
Child 2	0	1	1		1	0	0	1		0	0	1

Fig 3. Double Point Crossover

In our algorithm after each update to the pheromone amount we apply double point crossover if its value higher than the mean value of the global existing pheromones.. The pseudo-code for the modified ACO algorithm is as follows:

```
Modified ACO algorithm:
  Initialize parameters,
  initialize pheromone trails
  while termination condition not met do
  ConstructAntSolutions
  UpdatePheromones
  CrossoverPheromones
  If the Crossovered Pheromones gives a better solution
     Then UpdatePheromones
  end while
```

VI. Experimental Results

In this part we compare the results we got with basic ACO algorithm and Genetic algorithm approaches. For the purpose of comparing the performance of our modified algorithm we used a function that generates cities with the same coordinates.

Table 1. Comparing the results

Method	Cities	K Based Best Distance		
		K = 2	K=3	K=4
ACO	25	2396,55	2464,73	1873,02
Modified ACO	25	1551,28	1516,35	1603,41
Genetic Algorithm	25	1742,33	1621,51	1984,51
ACO	50	5233	4093,18	3329,23
Modified ACO	50	2006,59	2047,41	2201,19
Genetic Algorithm	50	2775,55	2283,82	2184,93
ACO	75	6207,52	4768,43	4228,98
Modified ACO	75	2278,76	2288,1	2216,37
Genetic Algorithm	75	3902,75	3390,4	2701,27

According to the experimental results shown in table 1 we see that the modified algorithm gives better results than the basic ACO algorithm and also give competitive results compared by the genetic algorithm.

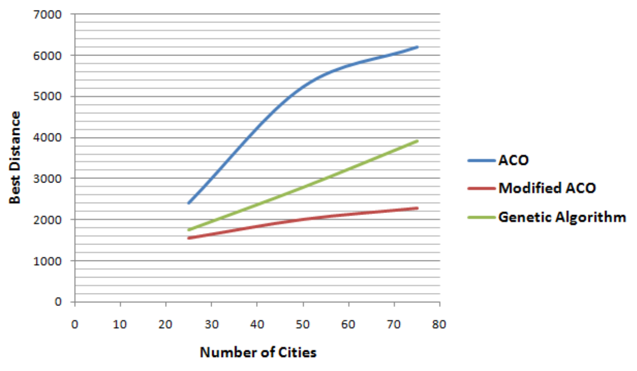

Fig 4.Comparison of the experimental results

In Figure 4 an instance of solving mTSP problem using the modified ACO algorithm with 40 cities and 4 salesmen where the big red points represent the correspondence centroid for each cluster.

The most competitive results were achieved by the modified ACO and the genetic algorithm approach. In Figure 4 we see an instance of solving mTSP problem using TSP. The instance includes 50 different cities and 3 salesmen.

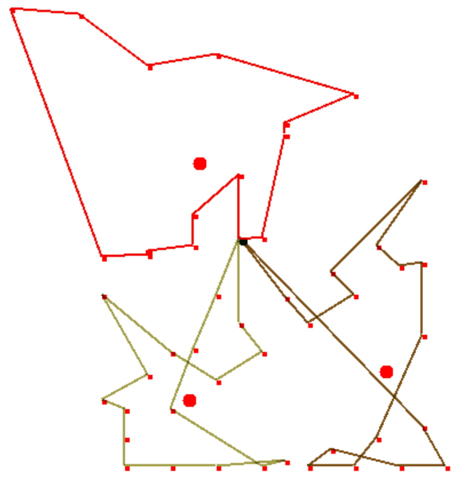

Fig 5. An instance of solving mTSP problem by the genetic algorithm approach

Whereas in Figure 5 we see the solution of the same instance by the modified ACO approach.

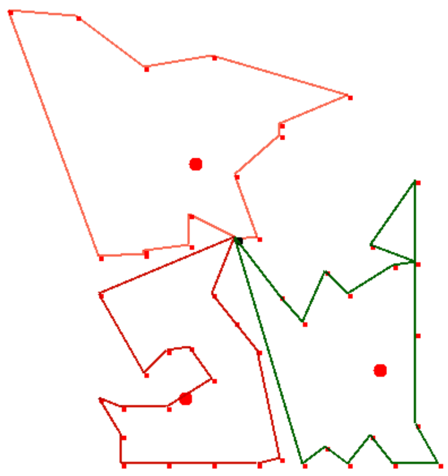

Fig 6. An instance of solving mTSP problem by the modified ACO approach

As a part of the test we also used 5 different TSPLIB instances to compare the performance of the modified ACO algorithm with other methods. TSPLIB is a library of standard instances for the TSP (and related problems) from various sources and of various types.[14]

According to the experimental results shown in table 2 we noticed that the modified algorithm gives at most the best minimum distances compared with both of ACO and genetic algorithm approaches.

Table 2. The computational results of benchmark problems

Method	TSPLIB instance	K Based Best Distance		
		K = 2	K=3	K=4
ACO	att48	71151,36	51032,81	42169,88
Modified ACO	att48	31585,43	28987,8	28510,82
Genetic Algorithm	att48	50725,81	49709,78	47083,53
ACO	berlin52	16354,02	11726,73	8820,24
Modified ACO	berlin52	7005,27	6409,53	6397,56
Genetic Algorithm	berlin52	11066,69	10898,75	11736,74
ACO	pr76	309514,32	265550,49	207333,11
Modified ACO	pr76	107261,01	100134,33	97405,06
Genetic Algorithm	pr76	184176,1	170857,76	168717,69
ACO	rat99	3980,43	3328,02	2814,74
Modified ACO	rat99	1301,58	1222,17	1210,58
Genetic Algorithm	rat99	2487,64	1970,48	1945,36
ACO	bier127	425016,29	349770,9	294273,77
Modified ACO	bier127	115048,5	115281,69	105762,11
Genetic Algorithm	bier127	282343,86	257228,63	233708,3

VII. CONCLUSIONS

In this paper, a new version of ant colony optimization (ACO) algorithm has been proposed for solving the mTSP problem. The modified algorithm is based on the crossover concept used in genetic algorithm approach. Our algorithm has been tested using the standard TSPLIB instances and compared with the basic ACO and genetic algorithm approaches and shows a good performance and also can be applied in different areas for the future use. The future work will be focus on combining the modified ACO with local search strategy in order to improve the solution quality.

REFERENCES

[1] N. Christofides, S. Eilon, An algorithm for the vehicledispatching problem. Operations Research Quarterly. 20,309-318, 1969.
[2] M. W. P. Savelsbergh, The general pickup and deliveryproblem. Transactions of Science. 29, 17-29, 1995 .
[3] Gorenstein S. Printing press scheduling for multi-edition periodicals. Manage Sci, 1970, 16: 373–383.
[4] Svestka J A, Huckfeldt V E. Computational experience with an msalesman traveling salesman algorithm. Manage Sci, 1973, 19: 790–798.
[5] Talay S S, Erdogan D R, Dept N. Multiple traveling robot problem: A solution based on dynamic task selection and robust execution. IEEE/ASME Trans Mechatronics, 2009, 14: 198–206.
[6] Qu H, Yang S X, Willms A R, et al. Real-time robot path planning based on a modified pulse-coupled neural network model. IEEE Trans Neural Networks, 2009, 20: 1724–1739.
[7] Russell R.A., An effective heuristic for the m-tour traveling salesman problem with some side conditions, Operations Research, 25(3), 1977, pp. 517–524.
[8] Majid YOUSEFIKHOSHBAKHT, Farzad DIDEHVAR and Farhad RAHMATI Modification of the Ant Colony Optimization for Solving the Multiple Traveling Salesman Problem
[9] Andrew Ng, The k-means clustering algorithm. http://cs229.stanford.edu/notes/cs229-notes7a.pdf
[10] https://en.wikipedia.org/wiki/Ant_colony_optimization_algorithms
[11] Dorigo, M., Maniezzo, V., and Colorni, A., 1996, 'The ant system: optimization by a colony of cooperating ants', IEEE Trans. Sys. Man Cybern. 26, 29–42

[12] Dorigo, M. and Gambardella, L. M., 1997, ‘Ant colony system: a cooperative learning approach to the traveling salesman problem’, IEEE Trans. Evol. Comput. 1, 53–66.

[13] D. NAGESH KUMAR and M. JANGA REDDY
Ant Colony Optimization for Multi-Purpose Reservoir Operation.

[14] TSPLIB: http://elib.zib.de/pub/mp-testdata/tsp/tsplib/tsplib.html

[15] Pan Junjie and Wang Dingwei, An Ant Colony Optimization Algorithm for Multiple Travelling Salesman Problem.

[16] Tim Kovacs :Relating Ant Colony Optimisation and Reinforcement Learning.

[17] Ali AI, Kennington JL (1986) The asymmetric m-traveling salesmen problem: a duality based branch-and-bound algorithm. Discrete Applied Mathematics 13:259–276

[18] Russell RA (1977) An effective heuristic for the m-tour traveling salesman problem with some side conditions. Operations Research 25(3):517–524

[19] Hsu CY, Tsai MH, Chen WM (1991) A study of feature-mapped approach to the multiple travelling salesmen problem. IEEE International Symposium on Circuits and Systems 3:1589–1592

[20] Zhang T, Gruver W, Smith M (1999) Team scheduling by genetic search. Proceedings of the second international conference on intelligent processing and manufacturing of materials 2:839–844.

[21] Shalini Singh and Ejaz Aslam Lodhi :Comparison Study of Multiple Traveling Salesmen Problem using Genetic Algorithm.

[22] Joel L. Ryan, T. Glenn Bailey, James T. Moore and William B. Carlton Reactive Tabu Search in unmanned aerial reconnaissance simulations.

[23] Stützle T, Hoos H (1997) Max-min ant system and local search for the traveling salesman problem. In: IEEE International Conference On Evolutionary Computation (ICEC’97), IEEE Press, Piscataway,NJ, pp 309–314.

[24] Marta S.R. Monteiro, Dalila B.M.M. Fontes Fernando A.C.C. Fontes: Ant Colony Optimization: a literature survey.