Journal of Engineering and Applied Sciences 13 (6): 1400-1406, 2018
ISSN: 1816-949X

An Ant Colony Algorithm with Dynamic Cities Allocation for Solving Competitive Travelling Salesmen Problem

[1]Mohannad Al-Kubaisi, [1]Belal Al-Khateeb and [2, 3]Muamer N. Mohammed
[1]College of Computer Science and Information Technology, University of Anbar, Ramadi, Iraq
[2]Faculty of Computer Systems and Software Engineering, University Malaysia Pahang, 26300 Kuantan, Pahang, Malaysia
[3]IBM Centre of Excellence, University Malaysia Pahang, 26300 Kuantan, Pahang, Malaysia

Abstract: In this study, an Ant Colony Optimization (ACO) algorithm is presented to address the Competitive Traveling Salesman Problem (CTSP). In CTSP there are a number of salesmen who aim to visit a number of cities. A salesman receives a benefit by visiting the city that has never been visited before. The overall pay off to a salesman is the aggregation of benefit earned by visiting cities minus the cost of the trip (travelled distance). The relationship between salesmen is non-cooperative as each salesman is working to increase their own benefit by visiting the largest possible number of unvisited cities. As it is difficult to find an optimal solution for CTSP an ACO algorithm is proposed. Inspired by the idea of real ant colony in which ants leave pheromone trails when looking for food in order to guide other ants to the target (food). To determine the number of ants a number of simulations on every problem is conducted. We find that 5 ants for 20 city CTSP and 125 ants for 300 city CTSP are good choices because they lead to high quality solutions. In this approach, all the cities are available for all salesmen at all the times. Each salesman will only choose its next city (according to his strategy) from the list of available cities to visit. Tests are carried out to measure the performance of the proposed algorithm and the obtained results suggest that ACO is a promising method for CTSP, since, it provide high quality solutions.

Key words: Travelling salesmen problem, competitive travelling salesmen problem, ant colony optimization, meta-heuristic, performance, simulations

INTRODUCTION

The Travelling Salesman Problem (TSP) is probably the most well-known combinatorial optimization problem (Adewole *et al*., 2012; Rani and Kumar, 2014). TSP is NP-hard, meaning that there is no known polynomial time algorithm that can return the optimal solution to any given instance. TSP is a generalized version of a Hamiltonian cycle where each node is visited exactly once (Shaikh and Panchal, 2012). In TSP, vertices on a graph represent cities, edges represent the path between cities and the value of an edge represents the distance between cities. The salesmen start from a city and must visit all cities once only and return to the first city with minimized cost (Firoozkooh, 2011; Taba, 2009). TSP emerged from Euler's studies "Studied the Knight's tour Problem" in 1766. In 1950, Dantzig, Fulkerson and Johnson found a method for solving TSP (Matai *et al*., 2010). They proved the effectiveness of their method by solving a 49 city instance. However, it became evident as early as the mid 1960's that the general instance of the TSP could not be solved in polynomial time using linear programming. Finally these categories of problems became known as NP-hard. Great progress was made in the late 1970's and 1980's when Grotschen, Padberg, Rinaldi and others managed to solve instances with up to 2392 cities, using cutting planes and branch-and bound (Matai *et al*., 2010; Demez, 2013). There are many variants of the TSP, including symmetric TSP asymmetric TSP, multiple TSP, competitive TSP and many others. The Competitive Travelling Salesmen Problem (CTSP) differs from the traditional TSP in that there is a competitive element between salesmen in visiting the cities. A salesman who is the first to visit a city will receive a payoff. All salesmen have to pay the costs of the journey (distance) whether they receive the benefit from being the first salesman to visit that city or not. The goal of each salesman is to visit as many cities as possible at the lowest cost. CTSP is a NP-hard problem where all salesmen have to take the strategy of other salesmen into consideration when planning their tours because of the conflict of interests (Kendall and Li, 2013; Li and Kendall, 2015).

Corresponding Author: Muamer N. Mohammed, Faculty of Computer Systems and Software Engineering, University Malaysia Pahang, 26300 Kuantan, Pahang, Malaysia

A real world example of CTSP is travelling circu's teams. Circuses travel from one city to another. When planning a trip, a circus has to take into account the distance traveled for the trip. Each circus also has to pay attention to the other team's paths to avoid the cities that have recently been visited by another circus. It is clearly a loss for the circus team if the cost of travelling is greater than the profit gained by visiting a city (Kendall and Li, 2013).

A CTSP is a complex non-cooperative problem where non-cooperation means that every salesman tries to increase their profit by trying to travel to the largest possible number of cities that have not been visited before. It is possible to calculate the Nash equilibrium on small instance of the CTSP but it is difficult to calculate the Nash equilibrium when the number of cities is large. Even if given all the routes for every salesman it is still an NP-hard problem. There is no known algorithm that computes optimal solutions for any given instance of an NP-Hard problem. CTSP is a dynamic problem that requires the salesman to plan the sequence if cities he/she will visit, taking into account the cost to visit, the payoff of visiting and the plans of the other salesman. These factors increase the complexity of the problem (Kendall and Li, 2013).

In this study, an ant colony optimization is presented to address the CTSP. Equilibrium analysis and other approaches that have been used for TSP are not suitable for CTSP. Ant colony optimization is a metaheuristic where artificial ants collaborate with each other to find reasonable solutions for the artificial ants, this group of artificial ants communicate with each other through the environment indirectly via. stigmergy. Good solutions are the result of collaboration feature between the artificial ants (Alp, 2004). The first ant colony optimization algorithm called ant system was proposed in the early nineties (Colorni *et al*., 1991).

Ant system, it has been providing many of the suggestions that are often different in the way of updating pheromone. In addition there are several additions suggested that differ from the ant system. Some of these variants are Elitist Ant System (EAS), Rank-Based AS, MAX-MIN AS, Best-worst AS and Ant Colony System (ACS).

Literature review: Colorni *et al.* (1991) proposed the first ant algorithm named Ant System (AS). It is a multi-agent approach, a category of distributed algorithms for combinatorial optimization. Like other ant type heuristics, the main characteristics of this heuristic was simulating or imitating the behaviour of a group of ants. These ants research cooperatively by using simple communication between them to solve an optimization problem. The first using of AS was solving the TSP. Despite encouraging initial results as could not compete with state-of-the-art algorithms for the TSP. Nevertheless, it had the important role of stimulating further research both on algorithmic variants which obtain much better computational performance and on applications to a large variety of different problems. Dorigo *et al.* (1991) described their methodology as a combination of distributed computation, positive feedback and constructive greedy heuristic. They applied their methodology to the classical TSP and the proposed system quickly provides very good solutions. Colorni *et al.* (1992) declared that this new approach can be used to solve any Combinatorial Optimization Problem (COP). They also state that a proper representation must be found as given below. The problem (a graph representation which is suitable for a search by many simple agents, the autocatalytic process, the heuristic rule that acts as a greedy force and allows a constructive definition of the solution, the constraint satisfaction method or tabu list. They apply AS to wide range of problems such as Satisfiability (SAT), Quadratic Assignment (QAP) and Job-Shop Scheduling (JSP) by using representation rules.

Brezina and Cickova (2011) studied a possibility of solving the TSP which ranges among NP-hard problems and offer a theoretical overview of some methods used for solving this problem. They discussed the Ant Colony Optimization (ACO) which belongs to the group of evolutionary techniques and presents the approach used in the application of ACO to the TSP. They studied the impact of some control parameters by implementing this algorithm. They compared the quality of solution with the optimal solution. Their experiments showed that the quality of solutions depends on the number of ants. The great advantage over the use of exact methods is that ACO algorithm provides relatively good results by a comparatively low number of iterations and is therefore, able to find an acceptable solution in a comparatively short time, so, it is useable for solving problems occurring impractical applications.

Fekete *et al.* (2004) introduced the Competing Salesmen Problem (CSP), a new variant of the classical traveling salesman problem. They suggested that multiple salesmen compete against each other to visit the largest number of cities instead of cooperating in finding the shortest route. At any given time, all salesmen know their opponent's positions. A salesman wins when he visits more cities than his competitors.

Kendall and Li (2013) have offered a different version of the traveling salesman problem in which a number of salesmen plan to visit a number of cities and the relationship between the salesmen are non-cooperative. The salesman gets a benefit if he visits a city that has not been visited by other salesmen. They have to pay the cost of traveling to a city. The goal of each salesman is to visit the largest number of cities that have not been visited before with minimum travel traveling distance. Each salesman needs to predict the tours of the competitors when planning his own tour. They suggested a hyper-heuristic algorithm to solve this problem. The hyper-heuristic algorithm consists of low level heuristics each of which can be used to build a tour and a high level heuristic which is used to choose among the low level heuristics at every decision point. One of big advantage the proposed system is the ability to inherit the features of low level heuristic to find the best solution. That show it can be developing much more efficient heuristic for CTSP.

Mohtadi and Nogondarian (2014) presented a game theoretic approach for solving the traveling salesman problem in competitive situations. They used game theory as a mathematical model to test the problems. They used genetic and tabu search algorithms. Experimental results show that the computational error is within a reasonable range. The tabu search algorithm generated solutions of better quality and less time was needed.

Li and Kendall (2015) presented a hyper-heuristic methodology to generate adaptive strategies for games. Based on a set of low-level heuristics (or strategies) a hyper-heuristic game player can generate strategies which adapt to both the behavior of the co-players and the game dynamics. They developed hyper-heuristic game players for three games, Iterated Prisoner's Dilemma (IPD), repeated Goofspiel and the Competitive Traveling Salesmen Problem (CTSP). They applied a hyper-heuristic for CTSP that has five low-level heuristics. Nearest Neighbor (NN), Random Neighbor (RN), Aggressive (AH), NN+2opt, RN+2opt. The high-level algorithm identifies the heuristics adopted by other agents and then selects from among its low-level heuristics. The computational results show that a. Hyper-heuristic game players outperform their low-level heuristics in repeated and dynamic games. Hyper-heuristic game players generate adaptive strategies even if the low-level heuristics are deterministic. Simple heuristic selection mechanisms can be adopted to construct automated game players in different games.

MATERIALS AND METHODS

Competitive travelling salesmen problem: Suppose that S = {1, ..., s} refers to a group of salesmen who are competing among themselves to visit N cities. The salesmen are selfish and they only care for their own payoffs. Therefore, it is a non-cooperative decision making problem. We will review the following conditions in CTSP.

Benefit: Every city $i \in \{1, \ldots, n\}$ has a fixed value of benefit Bi given to the first salesman to visit it and other salesmen receive nothing. If more than one salesman reaches a city simultaneously they share the profit among them equally.

Cost: Each salesman must pay the cost of moving from one city to another which can be calculated from the distance traveled. Let, C_{ij} denote the cost for an agent travelling from city i to city j.

Payoff: Can be calculated for each salesman through computed benefit received by the salesman in addition to calculating the cost of the transition to each city. Suppose that the salesman visited k cities and was able to get the benefit of cities k_0 ($k_0 \leq k$). The payoff is calculated by the following equation:

$$u = \sum_{j=1}^{k_0} B_j - \sum_{i=1}^{k-1} C_i \ (i+1) - C_{k1}$$

$k-k_0$ is the cities that have been visited by the salesman but was not able to get the benefit because they been visited by at least one other salesman.

Path: The salesmen are at the first located in different cities and they must travel from city to city. When the salesmen started a trip they cannot change their destinations (so, they have to define their entire route before they move to the first city). They must return to their departure city to complete their travel.

Speed of travel: For each salesman there is a constant speed during the trip.

Common knowledge: There is a set of common information available to every salesman this being the location, the speed of travel, the path travelled and the payoff for each salesman.

The goal for each salesman is to get the biggest payoff. In other words, each salesman aims to get a bigger benefit than the rest of salesman whilst at the same time minimising their travelling costs (Kendall and Li, 2013; Li and Kendall, 2015).

The solution of a CTSP is a Nash equilibrium(s) that denotes the state in which each agent cannot improve their payoffs by unilaterally changing their strategies.

However, it is not tractable to find the Nash equilibrium when the number of cities is large. The Nash

equilibrium has an effect on the agents in choosing their strategies only if it is known by all the agents involved. If some agents do not know the Nash equilibrium or they believe that some others do not know it, a Nash equilibrium is no longer a stable solution. Because of the difficulty in applying equilibrium analysis an ant colony optimization is introduced for the CTSP.

An ant colony optimization for CTSP: The ACS differs from the first ant system because of three main aspects:

- The state transition rule provides a direct way to balance between exploration of new edges and exploitation of a priori and accumulated knowledge about the problem
- The global updating rule is applied only to edges which belong to the best ant tour
- While ants construct a solution, a local pheromone-updating rule (local updating rule for short) is applied

All the cities are available for all salesmen at all the times. Each salesman will only choose its next city from the list of available cities to visit. The salesman who is firstly arrived at the city will get the benefit and the rest of salesmen are getting nothing. Therefore, the remaining salesmen avoid moving towards the city that visited by another salesman in order to avoid the loss.

Ant colony system for CTSP:

Initialization 1:

1. Get the dataset information (points), set No. of cities and compute the distances between cities according to the equation:

$$\text{Distance} = \sqrt{(y_2 - y_1)^2 + (x_2 - x_1)^2}$$

2. Arrange the cities in the site
3. Compute initial pheromone according to the equation:

$$\tau = (i, j) = \frac{1}{N \times \text{distance}(i, j)}$$

Where N: Number of cities.

4. Set the number of ants for each salesman

Solution 5:

5. Find start city for each salesman
6. Find first move for each salesman

For i = 0 to No. of Salesmen

Find next city(s) according to equation

$$j = \begin{cases} \max l \in N_i^k \left\{ \tau_{il} \left[\eta_{ij} \right]^{\beta} \right\} & \text{if } R \le q_0 \\ p & \text{otherwise} \end{cases}$$

where R is a random number in the interval [0, 1], $0 \le q_0 \le 1$ is a parameter of the decision rule called pseudo random proportional rule and p is the random proportional rule defined in following equation:

$$p_{r,s}^k = \frac{\tau(r, s)\mu(r, s)^{\beta}}{\sum_{u \in L} \tau(r, s)\mu(r, s)^{\beta}}$$

where k: is an ant, r: current city, s: next city, L: unvisited cites:

$$\tau: \text{phermone}, \mu = \frac{1}{\text{distance}}$$

Local update pheromone according to equation:

$$\tau(r, s) = (1-\rho) \times \tau = (r, s) + \rho\tau$$

Where:

$$\rho = 0.1$$

End for

7. Find the tour for each salesman and local update pheromone

No. of Cities = No. of cities>2

While (No. of cities>0)

Identify which salesman begin first according to travelled distance

Find next city(s) according to equation:

$$j = \begin{cases} \max I \in N_i^k \left\{ \tau_{il} \left[\eta_{ij} \right]^{\beta} \right\} & \text{if } R \le q_0 \\ p & \text{otherwise} \end{cases}$$

where R is a random number in the interval [0, 1], $0 \le q_0 \le 1$ is a parameter of the decision rule called pseudorandom proportional rule and p is the random proportional rule defined in following equation:

$$P_{r,s}^k = \frac{\tau(r, s)\,\mu(r, s)^{\beta}}{\sum_{\mu \in L} \tau(r, s)\,\mu(r, s)^{\beta}}$$

where k: is an ant, r: current city, s: next city, L: unvisited-cites:

$$\tau: \text{phermone}, \mu = \frac{1}{\text{distance}}, \beta = 2$$

Local update pheromone according to equation:

$$\tau(r, s) = (1-\rho) \times \tau \times (r, s) + \rho\tau$$

Where:

$$\rho = 0.1$$

No. of cities = No. of cities-1

End While

Return to start city

For i = 1 to No. of salesman

Tour = start city

Local update pheromone according to equation:

$$\tau(r, s) = (1-\rho) \times \tau \times (r, s) + \rho\tau$$

Where:

$$\rho = 0.1$$

End For

8. Global update pheromone

Update pheromone for edges that belong to best tour only

For i =1 to No. of salesmen

For j = 1 to No. of cities for each salesman

Global update pheromone according to equation

End For

End For

RESULTS AND DISCUSSION

An ant colony optimization algorithm is implemented and tested using a 2 salesmen 20 city CTSP and 3 salesmen 300 city CTSP. Two additional algorithms, nearest neighbour algorithm and random neighbour algorithm are implemented for the purposes of comparison.

Nearest neighbours algorithm (NR): This algorithm examines all the cities that have not visited before and chooses the nearest one.

Random Neighbours algorithm (RN): This algorithm will randomly select the next city from the list of unvisited cities.

In order to determine the best number of ants to be used, many experiments are conducted. The results of those experiments showed that five ants are good choice for 20 cities CTSP and 125 ants is good choice for 300 cities CTSP.

We have tested the influence of different parameters of ant colony. The experiment results show that changing these parameters does not have significant influence on the performance of the algorithm. Thus, the parameters were set to be $q_0 = 0.4$, $\beta = 2$, P1 = P2 = 0.1 in all the experiments for 20-city CTSP and $q_0 = 0.4$, $\beta = 4$, P1 = P2 = 0.1 will be used in all the experiments for 300-city CTSP.

In this approach, all the cities are available for all salesmen at all the times. Each salesman will only choose its next city (according to his strategy) from the list of available cities to visit. The salesman who is firstly arrived at the city will get the benefit from visiting this city and the rest of salesmen are getting nothing. Therefore, the rest of salesmen avoid moving towards the city that had visited by another salesman in order to avoid the loss. Three algorithms (NN, RN, ACS) will selected in order to compare the results at which each salesman adopts all the selected strategies. Therefore, the expected possibilities for two salesmen is nine and for three salesmen is 27 as shown in Table 1-4. The number of ants that used in the implementation of the 20 cities CTSP are five ants and for 300 cities CTSP are 125 ants. Payoff of visiting each city could be calculated by the following equation Payoff = Benfit-Cost. Where benefit consistently been identified 150 for each city. In addition, the costs are equal to the distance to move from one city to another. The results of those experiments are shown in Table 1-4. Results are two sets to both of these problems (20 and 300 cites CTSP) the best and the average of 1000 executions. The number of visited cities represent the number of cities that each salesman visited.

The results in Table 1-4 show that, the ant algorithm excelled in most of the cases that were in the face of other algorithms in both cases (best and average) for all experiments. This gives a good indication that ant algorithm is better than NN and RN which reflects a great success for the ant algorithm in solving the CTSP problem in the dynamic distribution of cities. In Table 1 and 2, ACS

Table 1: Best resultsof 1000 executions on 20 cities CTSP with two salesmen (parallel)

Algorithms	Payoff	The number of visited cities
NN vs. NN	(1395.1, 1312.79)	(11, 11)
NN vs. RN	(1493.73, 1087.66)	(12, 10)
NNVS ACS	(1395.1, 1312.79)	(11, 11)
RN vs. NN	(779.3, 1634.28)	(9, 13)
RN vs. RN	(1066.77, 1152.98)	(11, 11)
RNVS ACS	(736.59, 1826.68)	(8, 14)
ACSVS NN	(1395.1, 1312.79)	(11, 11)
ACS vs. RN	(1662.79, 870.56)	(13, 9)
ACS vs. ACS	(1395.1, 1312.79)	(11, 11)

Table 2: Average results of 1000 executions on 20 cities CTSP with two salesmen (parallel)

Algorithms	Payoff
NN vs. NN	(1344.9, 1339.99)
NN vs. RN	(1569.43, 932.68)
NN vs. ACS	(1344.9, 1339.99)
RN vs. NN	(932.62, 1566.84)
RN vs. RN	(1113.62, 1113.51)
RN vs. ACS	(931.37, 1568.19)
ACS vs. NN	(1344.9, 1339.99)
ACS vs. RN	(1568.42, 932.27)
ACS vs. ACS	(1344.9, 1339.99)

Table 3: Best results of 1000 executions on 300 cities CTSP with three salesmen (parallel)

Algorithms	Payoff	The number of visited cities
NN NN NN	(14841.58, 14557.1, 14115.22)	(103, 101, 99)
NN NN RN	(14953.23, 15184.2, 9422.53)	(104, 106, 93)
NN NN ACS	(14472.09, 14863.5, 14192.93)	(100, 103, 100)
NN RN NN	(15605.61, 9419.74, 14508.57)	(109, 93, 101)
NN RN RN	(20417.91, 8028.12, 8072.53)	(143, 80, 80)
NN RN ACS	(15071.27, 10244.9, 14995.09)	(105, 94, 104)
NN ACS NN	(14515.26, 15037.3, 14098.55)	(101, 104, 98)
NN ACS RN	(15064.25, 15149.35, 9890.51)	(105, 106, 92)
NN ACS ACS	(14539.53, 14570.92, 14141.94)	(102, 102, 99)
RN NN NN	(9957.51, 15245.33, 14933.34)	(93, 106, 104)
RN NN RN	(8189.01, 20800.15, 7446.43)	(79, 145, 79)
RN NN ACS	(8628.99, 15619.97, 14815.81)	(91, 108, 104)
RN RN NN	(8177.95, 7690.05, 20782.61)	(79, 79, 145)
RN RN RN	(9926.5, 9942.15, 10326.19)	(101, 101, 101)
RN RN ACS	(7694.69, 8046.84, 20905.32)	(80, 78, 145)
RN ACS NN	(8758.15, 15995.64, 14661.32)	(90, 111, 102)
RN ACS RN	(7570.45, 20867.87, 7647.68)	(79, 146, 78)
RN ACS ACS	(8946.03, 15276.42, 15274.36)	(90, 106, 107)
ACS NN NN	(14289.52, 15273.58, 13843.19)	(100, 106, 97)
ACS NN RN	(15866.89, 14840.4, 9407.88)	(110, 104, 89)
ACS NN ACS	(14694.43, 14522.92, 14335.32)	(102, 101, 100)
ACS RN NN	(14884.28, 9217.72, 15214.88)	(104, 93, 106)
ACS RN RN	(19977.24, 8695.84, 8161.27)	(141, 82, 80)
ACS RN ACS	(15333.78, 9254.98, 14442.22)	(107, 94, 102)
ACS ACS NN	(14273.19, 15180.44, 14009.78)	(100, 105, 98)
ACS ACS RN	(14796.03, 15546.19, 9086.62)	(103, 108, 92)
ACS ACS ACS	(14775.94, 14348.51, 14284.86)	(103, 101, 99)

Table 4: Average results of 1000 executions on 300 cities CTSP with three salesmen (parallel)

Algorithms	Payoff
NN NN NN	(14477.84, 14505.53, 14534.08)
NN NN RN	(15123.71, 15138.03, 9244.08)
NN NN ACS	(14452.79, 14471.9, 14604.94)
NN RN NN	(15150.78, 9249.95, 15091.87)
NN RN RN	(20716.53, 7908.76, 7890.15)
NN RN ACS	(15167.89, 9160.71, 15149.29)
NN ACS NN	(14455.08, 14589.75, 14478.94)
NN ACS RN	(15174, 15144.27, 9197.8)
NN ACS ACS	(14464.36, 14523.48, 14529.4)
RN NN NN	(9271.43, 15108.62, 15119.64)
RN NN RN	(7924.03, 20696.76, 7904.99)
RN NN ACS	(9215.46, 15163.86, 15138.51)
RN RN NN	(7915.84, 7897.08, 20682.47)
RN RN RN	(9858.63, 9875.84, 9839.3)
RN RN ACS	(7908.71, 7909.17, 20644.64)
RN ACS NN	(9209.38, 15173.44, 15115.77)
RN ACS RN	(7935.02, 20625.31, 7908.84)
RN ACS ACS	(9202.24, 15193.05, 15121.77)
ACS NN NN	(14545.19, 14487.18, 14494.91)
ACS NN RN	(15154.55, 15172.2, 9217.69)
ACS NN ACS	(14514.54, 14483.84, 14522.29)
ACS RN NN	(15154.45, 9184.52, 15150.92)
ACS RN RN	(20663.43, 7913.1, 7887.68)
ACS RN ACS	(15203.69, 9161.53, 15114.39)
ACS ACS NN	(14450.26, 14549.39, 14505.35)
ACS ACS RN	(15225.73, 15137.47, 9134.52)
ACS ACS ACS	(14515.64, 14505.04, 14475.73)

algorithm is the best in four out of five times also, NN algorithm is the best in four out of five times and RN algorithm is the best in one time only out of 5 times. In Table 3 and 4, ACS algorithm is the best in 15 out of 19 times, NN algorithm is the best on 12 out of 19 times and RN algorithm is the best in 1 time only out of 19 times. Those results gave a lot of confidence in the ACS algorithm as best suited to solve CTSP as there are rare cases that the ACS algorithm is overtaken by the other two algorithms.

CONCLUSION

CTSP is a hybridization of a decision making problem and an optimization problem. Through the strategic interaction among Salesmen in the selection of tours as well as the payoff calculation, the problem can be considered as an example of a realistic decision-making processes. It is very difficult to find an optimal solution to CTSP. In this study, we have proposed ant colony algorithm to address the CTSP. Selecting the number of ants is important as it has a significant influence on the performance of the algorithm. An experiment to choose the appropriate number of ants that provides the best trade off (for-profit and time) has been conducted. We find that five ants provide the best trade-off for the 20-city CTSP and 125 ants are most suitable for 300-city CTSP. We designed and implemented an ant algorithm to solve the CTSP in addition to other two algorithms (NN and RN) for comparison. In the proposed approach all the cities are available to all salesmen at all time and each salesman can choose the next city according to his strategy. We found that the ant algorithm generated very good results, compared with other algorithms in a reasonable time as the local and global update of the pheromone help ants to avoid visiting the cities that do not bring benefit.

The success of ant algorithm suggests the potential of some other algorithms to solve the CTSP such as using other nature inspired algorithms like bees algorithm as well as possible applications of population algorithms (such as genetic algorithm, tabu search algorithm, scatter search algorithm, etc.) and others.

REFERENCES

Adewole, A.P., K. Otubamowo and T.O. Egunjobi, 2012. A comparative study of simulated annealing and genetic algorithm for solving the travelling salesman problem. Int. J. Applied Inform. Syst., 4: 6-12.

Alp, A., 2004. Ant colony optimization for the single model U-type assembly line balancing problem. Ph.D Thesis, Bilkent University, Ankara, Turkey.

Brezina Jr, I. and Z. Cickova, 2011. Solving the travelling salesman problem using the ant colony optimization. Manage. Inf. Syst., 16: 10-14.

Colorni, A., M. Dorigo and V. Maniezzo, 1991. Distributed optimization by Ant Colonies. In: Toward a Practice of Autonomous Systems: Proceedings of the European Conference on Artificial Life, Varela, F.J. and P. Bourgine (Eds.). MIT Press, Cambridge, Massachusetts, USA., pp: 134-142.

Colorni, A., M. Dorigo and V. Maniezzo, 1992. An Investigation of Some Properties of an Ant Algorithm. In: Parallel Problem Solving from Nature, 2: Proceedings of the Second Conference on Parallel Problem Solving from Nature, Manner, R. and B. Manderick (Eds.). Elsevier Publishing, Brussels, Belgium, pp: 509-520.

Demez, H., 2013. Combinatorial optimization: Solution methods of traveling salesman problem. Ph.D Thesis, Eastern Mediterranean University, Famagusta, Northern Cyprus.

Dorigo, M., V. Maniezzo and A. Colorni, 1991. Positive feedback as a search strategy. Technical Report No. 91-016, Politecnico di Milano, Italy.

Fekete, S.P., R. Fleischer, A. Fraenkel and M. Schmitt, 2004. Traveling salesmen in the presence of competition. Theor. Comput. Sci., 313: 377-392.

Firoozkooh, I., 2011. Using imperial competitive algorithm for solving traveling salesman problem and comparing the efficiency of the proposed algorithm with methods in use. Aust. J. Basic Appl. Sci., 5: 540-543.

Kendall, G. and J. Li, 2013. Competitive travelling salesmen problem: A hyper-heuristic approach. J. Oper. Res. Soc., 64: 208-216.

Li, J. and G. Kendall, 2015. A hyper-heuristic methodology to generate adaptive strategies for games. IEEE. Trans. Comput. Intell. AI Games, 9: 1-10.

Matai, R., S. Singh and M.L. Mittal, 2010. Traveling Salesman Problem: An Overview of Applications, Formulations and Solution Approaches. In: Traveling Salesman Problem, Theory and Applications, Davendra, D. (Ed.). InTech Publisher, Rijeka, Croatia, ISBN:978-953-307-426-9, pp: 1-24.

Mohtadi, M. and K. Nogondarian, 2014. Solving the traveling salesman problem in competitive situations using the game theory. Appl. Math. Eng. Manage. Technol., 2: 311-325.

Rani, K. and V. Kumar, 2014. Solving travelling salesman problem using genetic algorithm based on heuristic crossover and mutation operator. Intl. J. Res. Eng. Technol., 2: 27-34.

Shaikh, M. and M. Panchal, 2012. Solving asymmetric travelling salesman problem using memetic algorithm. Int. J. Emerg. Technol. Adv. Eng., 2: 634-639.

Taba, M.S.E., 2009. Solving traveling salesman problem with a non-complete graph. Ph.D Thesis, University of Waterlo, Waterloo, Canada.