University of Salford MANCHESTER

@00311262

MSc Robotics and Automation
School of Computing, Science and Engineering

MSc Dissertation

ANT COLONY ALGORITHM AND GENETIC ALGORITHM FOR MULTIPLE TRAVELLING SALESMEN PROBLEM

Author: BHARATH MANICKA VASAGAM

Supervisor: Dr. A.H. Jones

2012

CONTENTS

LIST OF FIGURES

LIST OF GRAPHS

Graph 1: Pareto plot between the no. of cities VS the best solution of GA and best solution of ACO for TSP

ACKNOWLEDGEMENT

Huge thanks to my supervisor Dr A.H.Jones for providing me great insight and knowledge, for his continuous encouragement and for providing me with many of his valuable time.

I thank you so much sir for giving me such an opportunity to work under you. I thank my parents who continued to support me, morally and financially. I would also thank my friends who gave me moral support and stayed at university with me for many days. Without their support I would not be able to put so much time and effort into my courses and projects.

Finally I thank goes to the University of Salford, Robotics and Automation department staffs for their support and taking me to a next level in my studies.

Last but not least, I thank the Almighty, God for his blessings which helped me to complete my project.

ABSTRACT

Ant Colony Optimization (ACO) is a meta-heuristic for solving combinatorial optimization problems. Among the social behavior inspired algorithms, Ant colony algorithm has proved to be better one.

In this master thesis, I propose a comparison between ACO algorithm and Genetic algorithm in solving Multiple Travelling Salesmen Problem. The main objective of the algorithms is to minimize the cost function. The design of algorithm is to be made by MATLAB software and the results are analyzed using the same.

These two algorithms and their different variants have been studied and tested on many conditions. They have been compared through the analysis of more than one graph and different configurations. The results obtained have helped us to have a better understanding of the problem and provide indications and suggestions for further research.

1.0 INTRODUCTION:

Ant Colony Optimization is a meta-heuristic for solving problems involving optimization. This technique of Ant colony Optimization was inspired by the behavior of ants in finding their short path from their living place to food.

The unique feature in the ant colony is that, it finds the optimized path towards its food without any outer centralized control or direct communication within the ants. This is the basic concept of swarm intelligence that is observed not only in ants, it can be seen in birds, whales etc. The communication between the ants is based on the deposit of the pheromone throughout their path. The variation in the amount of pheromone deposited in the path taken by ants will indicates the quality of the route selected to reach the food. This idea was first found and ACO algorithm was framed by Dorigo in 1992 [1].

After this first invention, there evolved many variety of classical hard combinatorial optimization problems like Travelling Salesman Problem, clustering problem, Multiple Travelling Salesman Problem and many other applications. Many researchers had performed a comparative study of various combinatorial optimization techniques like Genetic Algorithms, Particle Swarm Optimization etc. Ant Colony Optimization has proved to be a better solution for some multi objective optimization problems [2].

Travelling Salesman Problem is one of the NP hard problems and is generally considered the most highly studied problem in computational mathematics. [3]. One of the best-known solutions for solving the TSP is with the application of the Evolutionary algorithms. These algorithms are basically developed on naturally occurring principles in nature. Many algorithms prevail like Ant Colony Algorithm, Genetic Algorithm, Particle Swarm optimization and many more. There are algorithms also based on Differential evolution, Scatter search and Tabu Search which has also been proven to be good at solving the TSP [4].

The importance of travelling salesman problem is seen in many practical applications of life. For example, if a person has to deliver goods to several parts of a city, he must first decide the best route or the order of places to be visited to save his time and energy. At higher level this is the base for a robot which tries to decide its own path of exploration in an unknown

environment. There has been a lot of literature for TSP whereas Multiple Travelling Salesman Problem received a lot less attention from the researchers [5].

The Multiple Travelling Salesman problem (mTSP) is a type of TSP with more than one objective to be solved. The characteristic of the mTSP is more likely to have a lot of real time applications.mTSP finds its application mainly in UAV (Un-manned aerial vehicle) path planning, manufacturing routing such as movement of parts along manufacturing floor or the amount of solder on circuit board and Network design such as determining the amount of cabling required [6]. The mTSP has not got the attention of researchers as the TSP.

1.1 AIM

In this Master thesis, we will propose an ant colony optimization for a multi objective combinatorial problem i.e., Multiple Travelling Salesman Problem and compare its effectiveness with Genetic algorithm that solves the same problem.

2.0 LITERATURE REVIEW

2.1 SINGLE TRAVELLING SALESMAN PROBLEM:

The Travelling Salesman Problem (TSP) is one of the widely researched optimization problems. The problem statement is quite simple but it still remains as the most challenging problem. The objective of TSP is finding the minimum distance circuit through a set of vertex by reaching each vertex only once. This minimum distance circuit is called Hamiltonian circuit. In some cases the minimum distance could be replaced by minimum cost or minimum travel time. The most common practical explanation of TSP is that a salesman trying tofigure out the shortest tour between *n* cities [62]. Some permutation problem that is not directly associated with the routing problem can also be referred as TSP. Some common applications of TSP are

i. Wiring of Computer: [63] Some computer systems which were at earlier stages were as modules and pins. It is required to link the pins with series length of wires. The length of the wire must be minimized.
ii. Punching Hole: [64] In some manufacturing process, it will be required to punch hole on metallic sheets. The objective is to have minimum number of punches and

maximum holes in it. Such a problem occurs in circuit board manufacturing industries. Problems are of large scale.

iii. Simple job sequencing: If there are *n* jobs and it must be performed in a single machine with minimum change over time, the best sequence is figured out by TSP.

iv. Design of Dartboard: [65] Dartboards is a game with circular targets. It has concentric circles and 20 sections from 1 to 20. The aim of the game is to reduce the total value 301 to zero by scoring the values on the board. The application of TSP here is to design the dart board with scores from 1 to 20 depending on the toughness of the game.

v. Crystallography technique:[66]: The X-ray intensity measurements on a crystal is taken by means of detector for some experiments. This is called crystallography. The detection requires the sampling of the crystals and the placement of crystals in detector appropriately. The order of measurement is the solution of TSP. The applications are very high and if the best solution are found, then it would be greatly reduce the time taken to undergo these measurements.

2.2 MULTIPLE TRAVELLING SALESMEN PROBLEM:

Multiple travelling salesmen problem is the generalization of the TSP. In TSP, there is one salesman and several cities and objective is to find the best path with the minimal cost. But in mTSP there are *m* salesmen and each salesman visits a set of cities, so that all cities are visited. The objectives here in mTSP are that each salesman has got a limited no of cities to visit for which a minimal route is to be determined and the overall distance i.e., sum of distances travelled by each salesman should also be minimum [7]. The mTSP has got more combinatorial complexity because of multi-objective nature. Generally the mTSP problem can be explained as follows: Let *n*, be the no of cities and *m* be the no. of salesmen The Salesmen can depart from any depot and reach their own respective depots only once with the minimized cost function. The cost function here represents the minimum time and distance. This is one type of the mTSP. The application mTSP can be varied according to the requirement. The salesmen can choose any depot in random or they can all have a single depot. There is also an option of salesmen not

required to return to their own depot which is called the open ends mTSP [8]. So all these solely depends upon the practical application in which the mTSP is going to be employed.

There are certain variations of this mTSP as follows.

1. Single and multiple depots: In the single depots, the entire salesman would start from a single city and end up in the same city. In multiple depots, all salesmen would start at a different random cities and end up in the city where they started. Even in the multiple depots there are some variations. If there exist multiple depot, the salesmen would need to return to their destinations in fixed destination case [9]. In non fixed destination case the salesman would end up in any city.
2. Number of salesmen: The number of salesmen in a problem is fixed [9].
3. Time window: The most important variation of mTSP would be the time windows. Certain nodes are to be travelled in definite time periods called time windows. The mTSP extension is called the multiple travelling salesman problems with time windows [9]. This extension has a lot of applications like airline scheduling.
4. Other criteria: There are some other variations like number of minimum nodes that each salesman should have visited or the minimum distance that must be travelled by each salesman [9].

The mTSP can be considered as a type of Vehicle Routing Problem with lesser restrictions. The single travelling salesman problem is also a type of mTSP. If the number of salesman in mTSP is 1 then it is reduced to TSP which has a lot of research and extensive algorithms to solve the problem have been formulated.

mTSP APPLICATIONS:

There are many applications of this mTSP problem,

i. Scheduling of Crew: An application for a deposit which is to be carried between the different branches of the bank is considered as a problem [10]. The deposit is picked up from local banks and is to be returned to the main branch. The crew members will have to collect the deposits. The problem here is to determine the routes of the crew

members with a minimum cost. Cost, represents the energy and time. There were several applications reported by [11] and [12].

ii. Planning of Mission: The planning of mission in an autonomous robot arises in variety of places like warehouse automation, planetary exploration etc.If there are ‘m’ robots and ‘n’ goals for the robots, the robots must visit all the goals and then return to the common depot. The main application of mTSP is the effective planning of the pathway [13] [14].

iii. Global Navigation Satellite System: This system is to hold surveillance on the global management, forthcoming disasters, and agricultural practices etc. There will be multiple receivers placed all over the geographical points. The application of mTSP helps in choosing the best set of receivers for a particular session [15].

iv. Scheduling of print Press: The problem of scheduling of print press of periodicals publications with multi-editions was explained in two papers [16] [17]. There are five pairs of cylinders through which the paper rolls and both sides are printed at once. There are several types of 4-, 6-, and 8- page forms to be printed. The application of this mTSP arises on which form of page to print and the length.

v. Hot rolling installation problem: The scheduling of orders on hot rolling mill installation is an application of mTSP [18]. The number of orders is considered as the number of cities and the fine cost to be paid for the replacement of orders was the distance between cities.

vi. Scheduling of Interview: In interview scheduling of tour brokers and vendors in some tourism industry there is application of mTSP. Broker corresponds salesman, the vendors to be visited corresponds the cities. This application was found and solved [19].

vii. Routing of School bus: The application of mTSP on school bus routing is obvious. The pattern of loading the school bus is to be found. The distance is kept least and the application was experimented successfully [20].

3.0 ALGORITHMS FOR mTSP:

The solution to this mTSP was first given [21] but by ignoring some of the constraints the mTSP had. The only problem they considered was the fixed cost that was incorporated with each salesman. The sub tour elimination (SEC) constraint was ignored initially in the algorithm and then later a normal check was performed whether any SEC are violated after the solution was formed [21]. The mTSP was solved to an optimal solution [22]. The algorithm followed the branch-and-bound method. The degree constrains were relaxed and the Lagrangean problem was followed. The Lagrangean problem can be solved using a minimal spanning with degree constraint. The results of this optimization showed that the Lagrangean problem decreases when the problem size was increased with number of cities more than 400. Another algorithm which was based on QA relaxation (Quasi assignment) was found [23] and the solution was attained by ignoring the SEC and solving the QA in polynomial time.

3.1 APPROACHES FOR APPROXIMATION:

There are two methods to solve mTSP: Exact approach method and Heuristic method. The heuristic algorithms would take a long computation time and thus many such methods have been developed to reduce the computational time or computational complexity. The best solution with a good algorithm was described [24]. The complexity of such approximation algorithm is O $(n\ (\log_2 n)^{O(c)})$ and 'n' represents the problem size. The solving processof mTSP can be split into parts like Initial Tour construction and Tour improvement until optimum solution is reached.

(i) Initial Tour construction methods:

The construction of tours is done until the initial solution is formed. The tour constructions help to find the solution which is considered as 10 – 15% optimum [24]. Few of the many such initial solution construction methods are as follows.

(ii) Nearest Neighbour Algorithm:

This the basic initial solution construction method used. The idea of this method is to find the nearby city for each salesman. The complexity of this approach is given as $O(n^2)$. The steps involved in this algorithm are,

1. Selection of a random city for each salesman
2. Computing the nearest city that is unvisited and to move there
3. If there any city left unvisited, repeat the step 2
4. Return to the initial depot (first city).

In fact this method account for 25% optimality of the solution and this the highest of any other methods [25].

(iii) Greedy Heuristics:

The greedy heuristics [26] is applied in a way that shorter edges are found and added to each salesman until it does not create a tour with less the required N edges. It is not legal to add any edge twice. The complexity of this method is O ($n^2 \log_2 (n)$). The steps involved in this approach are

1. Arrange all the edges
2. Select the shortest edge to each salesman if it does not violate any constraint.
3. Do all tours contain the minimum N edges in its tour? If not, repeat the step 2.

The optimality to the solution reached by this greedy algorithm is about 15-20% reported by (s).

(iv) Insertion heuristic method:

The insertion method [26] is also straight forward as nearest neighbor algorithm. The complexity is also same as O (n^2). An initial subtour is formed which is mostly a triangle. Then the rest of the tours are inserted by some heuristics. The steps involved in this process are

1. Selection of the shortest edge and a subtour is made.
2. Selection of city not included in the subtour having the shortest distance with anyone of the city in subtour.

3. Searching a suitable edge where the cost of insertion would be minimal
4. Repetition of step 2 until all the cities is covered.

3.2 TOUR IMPROVEMENT:

Once the initial solution is found, then a tour improvement heuristic is applied to improve quality of solution. The most populous tour improvement heuristics are 2-opt and 3-opt which solely depends upon the initial tour construction [27]. There are also some meta-heuristic approaches available like tabu search, evolutionary algorithms, simulated annealing etc.

(i) 2-opt & 3-opt algorithm:

The 2-opt algorithm takes any two edges from an initial constructed tour and new two paths are re-inserted. This is the 2-opt movement. While re inserting the new route it happens in such a way that tour is still valid. The re-insertion is done only when the new route is shorter than the old ones. This is continued until no further optimization is possible [27]. The final solution is now called 2 optimal as shown below.

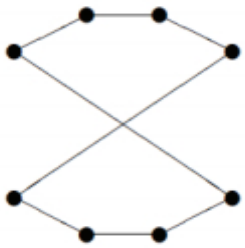


Figure 1: 2-opt algorithm

The working principle of 3-opt algorithm is same as that of a 2-opt algorithm. But the only thing that varies is the number of path re-inserted. 3 edges are removed instead of 2. The algorithm stops when no more 3-moves are possible. If a tour runs 3-optimal it is also said as 2-optimal [27].

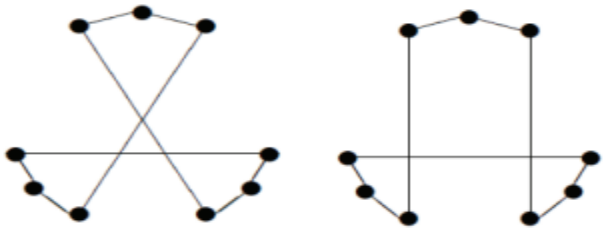


Figure 2: 3-opt algorithm

(ii) k-opt algorithm:

This k-opt algorithm can be used to improve the solution further than that of a 2-opt or 3-opt algorithm would do. The 2-opt and 3-opt algorithms are a type of k-opt where the k=2 and 3 respectively. K-opt can be naturally used having k>3 and it takes a lot of time to optimize [27]. The 4-opt move can be called as crossing bridges as shown in the program.

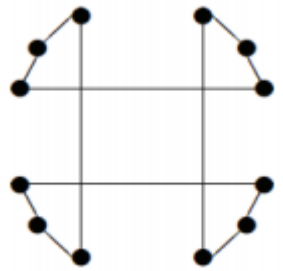


Figure 3: 4-opt algorithm

(iii) Tabu Search:

It is a type of neighborhood search which searches the better solution in neighborhood of initial solution. This search uses both 2-opt and 3-opt search together which searches for a best solution. But the greatest disadvantage of this search is that it stops at local minima. The methods of implementing a good tabu search and ways of avoiding the problem converging at local minima is explained [28].

(iv) Genetic algorithm:

Genetic algorithm is a nature based algorithm. Firstly a random population is generated. Few of the population selected to produce the next child and then the mutation process occurs. The fitness value of each child determines how good it is. Then for the next generation most fit

candidates are chosen. And when further mutation occurs the fitness of the next generation would increase [29]. There are several routines in application of GA like, crossover routine, measure of fitness routine, and a mutation routine. Good solution is indicated by the best fitness value. There are different ways of mutation and crossover accounted by different researchers to find better solution in good time.

(v) Ant Colony Algorithm:

The ant colony algorithm is an idea to replicate the ant movements. This idea was successful in giving a good solution to TSP very quickly [30]. Ants leave out a trail of pheromone when they explore new region. The next ants would follow the same and the trail becomes strong since many ants follow the same path. Some ants are placed in each city and they form the initial solution by some heuristic. Then the ant which followed the shortest route will have the best pheromone. Next when a solution is chosen this best pheromone trail is taken into account and this process is continued till the best solution is formed.

(vi) Other Heuristic approaches for mTSP:

A heuristic based on exchange procedure to solve mTSP was given by [31]. An approach called parallel processing with the usage of evolutionary algorithms was proposed [32]. It has two salesmen. The objective function was to minimize the difference in route lengths of routes for both salesmen. When the problem exceeds having more than 25 to 50 cities, evolutionary algorithm has proved to obtain optimal solution. The ANN (Artificial Neural Networks) has also proved to be better in solving the mTSP. But they are just the extended version of ones used in TSP. The extended Hopfield-Tank ANN model was used to solve mTSP [33]. But later this method was considered too complex and was proved that it fails to provide the feasible solutions. Neural network approach was also used to solve mTSP [34]. Even this method was also a bit of extension of TSP. It solved the m standards of normal TSP. The results were better than that of Hopfield tank ANN model. Another approach based on the elastic net was found to solve mTSP [35]. This is a self-organizing neural network approach. The recent approaches which were found to better than the elastic approach was developed [36] and [37]. The objective function minimizes the cost of expensive route of each salesman.

3.3 GENETIC ALGORITHM FOR mTSP

Genetic algorithms were first used [38] and the solution was developed using the genetic operators. A modified genetic approach based on the conversion of mTSP into a single TSP and then solving was found [39]. The genetic algorithm to solve mTSP and then applying in the problem of path planning was also considered [40]. By taking into consideration the time windows, Tabu search was used to solve mTSP [41]. The application of a class of neighborhood search technique called cyclic transfers to mTSP problem was also given [42]. Competition based NN was suggested [43] to solve minmaxmTSP. A technique called simultaneous generalized hill climbing algorithm (SGHC) was used to solve mTSP[44]. Genetic Algorithm was found to have large application areas and since then, it has been usedby many to solve mTSP with minor alterations in the algorithm. In addition a hybrid genetic algorithm was proposed, which was based on the tabu search for solving mTSP [45].

As discussed in previous paragraph due to combinatorial complexity of mTSP, heuristics needs to be applied to solve the problem. GA which is one of the heuristics can be applied to TSP problem and remarkably researched. The researchers studying the Vehicle Routing problem researched that GA developed to solve the TSP can also be used to solve the mTSP. GA was first used to solve TSP [46]. The solving of mTSP using GA has focused on the vehicle routing problem [47] [48]. The problem of vehicle scheduling consists of *m* vehicles to be scheduled to visit *n* cities. Each city must be visited only once by each vehicle. There are different configurations of this vehicle scheduling problem; VSP without or with time windows, VSP with homogenous or heterogeneous vehicle, VSP for minimum vehicle or minimum distance. So all these type leads to various types of objective.

Genetic algorithms are new type of optimization method that can be applied to combinatorial problems. The basic idea of GA was found [49]. GA was not developed to solve the optimization problems. But it was later adopted towards its solving the problem [50][51]. Then GA was greatly used to solve TSP. The performance of GA was improved by designing the good GA operators [52].

GAs generates a population of numeric vectors (chromosomes) which represents a solution of the problem. The separate components of the chromosome is called gene. New

chromosomes are produced at the next generation by the process of crossover (probability of exchange of values between the vectors) or by mutation (random replacement of the values in vector). Mutation which operates by random exchange increases the search space coverage and prevents the earlier convergence of the solution [53]. The Chromosomes are evaluated with the help of the fitness function which is the least distance in case of mTSP. The fitness function helps the fittest chromosome to survive and those that are less fit are eliminated. Thus as a result of this cross over and mutation, the gene pool gets improving into better and better, thus a better solution is got [53]. The process of GA continues until the specified fitness value is got, or if no improvement occurs for specific iterations.

The main idea of getting a good solution out of this GA depends on the chromosome representation of the problem. Good chromosome representation must eradicate the redundant chromosomes. The redundant chromosomes are the one occurring often in the solution multiple times and it is representation of a better solution being able to be represented in more ways [53]. Thus the redundant chromosome will decrease the search speed and increases the search space.

A good chromosome will denote the solution and allows the GA operators to work on it to generate better offspring as the evolutionary iterations progress. Next the importance is given to the operators which process the good chromosomes. For example in TSP, the nature of the solution is that, salesman should visit the cities only once and without sub-tours. Thus the evolution occurs and the feasibility of the solution is also maintained [53]. So the crossover methods will change the solution.

A huge number of cross over methods have been proposed for solving TSP using GA. Some of the crossovers are Order crossover, Partially mapped crossover, Cycle crossover, Asexual crossover.

3.1.1 Chromosome representation:

There are two different chromosome representations for the mTSP using GA.

Two Chromosome technique:

As the name suggests, there are two chromosomes. The example below shows that there are '*n*' numbers of cities permutated in chromosome 1, while the next chromosome contains the assignment of salesman to each of the cities in the corresponding position of first chromosome [47].

Cities

5	2	14	1	6	8	13	1	10	4	3	12	15	7	9

Salesman

1	2	1	1	4	3	2	1	1	2	3	3	4	2	1

Figure 4: Example of 2 chromosome representation for 15 cities and 4 salesmen

In the example shown in Figure 4, cities 2,13,4 and 7 are visited in order by salesman 2. It is in the same manner as that of salesman 2 for other salesmen too. Now using the above two chromosomes there are $n!m^n$ possible solutions for the problem where n is number of cities and m is the salesman number. Many of those solutions are redundant. Some of the gene in the above shown chromosome could be interchanged to form an identical solution [47].

One chromosome technique:

5	2	14	1	-1	6	8	13	1	-2	10	4	3	-3	12	15	7	9

Salesman 1 Salesman 2 Salesman 3 Salesman 4

Figure 5: Example of 1 chromosome representation for 15 cities and 4 salesmen

The figure 5 illustrates the chromosome representation of one chromosome technique. The length of chromosome is n+m-1(For n=15 an m=4). In this technique, n cities are permutated from 1 to n. The permutation is partitioned into m-sub tours (-1 to – (m-1)) that represents the change from one salesman to other. First salesman would visit 5, 4, 14 and 1 in this order and so on. Using the chromosome structure, there is (n+m-1)! possible solutions. Many of the solutions are redundant. There is a possibility of getting two or more negative numbers consecutively which helps in reducing the number of salesman. So the salesmen would be used effectively [18].

Two-part chromosome technique:

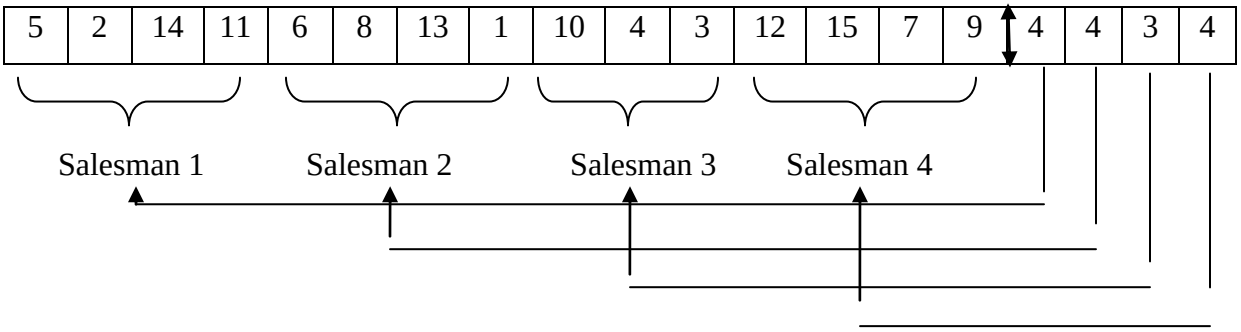


Figure 6: Example of 2-part chromosome representation for 15 cities and 4 salesmen

This is another type of chromosome representation. This method reduces the size of the search space due to the elimination of redundant solutions. Cities that are assigned to salesman 1 always appear in the first followed by the set of cities assigned to the next salesman. There are n! possible permutation of the first part of chromosome. There are $\binom{n-1}{m-1}$ distinct positive integer-valued m-vectors which satisfies the requirement [54]. Therefore the solution space for this technique is of size $n!\binom{n-1}{m-1}$

Here in this thesis an ant colony algorithm was designed to solve mTSP and its effectiveness was compared to that of the Genetic algorithm.

4.0 ANT COLONY OPTIMIZATION

Origin:

The ACO algorithm is developed or inspired from the real ants' behaviour when they look for food. In real time, the ants look for their food and they explore the unknown environment randomly. The ants deposit the pheromone trail while travelling. When they get food more ants are attracted towards the strong pheromone trail. Thus a mass of ant will get through the strong pheromone route than the weaker ones. As time goes, the route which attracts lesser ants implies that the pheromone has got evaporated. The probability of ant choosing a long route will get lowered as the time increases. An experiment to demonstrate the self-organization of ants because of the pheromone trails was demonstrated by Binary bridge experiment [55].

An ant colony is separated from a food source with binary bridge. The ants are left with two choices to reach the food. The two paths A and B are of same length as shown in the Fig. As the ant begins to explore the food they choose either of the two routes in random. So the probability of choosing the either route was 50% at the initial stage.

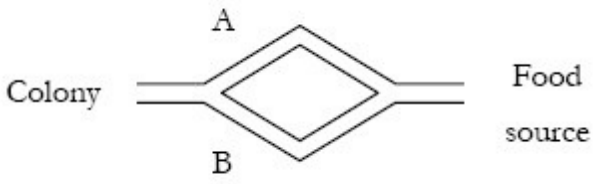


Figure 7: Binary bridge

At initial stage there is no factor that influences the decision of the ant to choose either of the routes. Each ant travelling through the routes lays the pheromone trail at a constant rate to and from the source of food. Nearly equal ants pass through both the routes at the initial stage of the experiment. But after some time due to some fluctuations in the randomness, density of ants passing through one of the routes gets increased and the pheromone trail in those routes is with high intensity [55]. And gradually all the ants starts using the single route. As the relative

intensity of the pheromone across one route increases, there is a very low probability for an ant to choose the other route with lesser pheromone.

If P_A , probability of ant *i+1* choosing either branch is to be calculated from A_i and B_i number of ants of branch *A* and *B* respectively. So the probability PA of the i+1 th ant to choose the branch A is

$$P_A = \frac{(k + A_i)^m}{(k + A_i)^n (k + B_i)^n} = 1 - P_B$$

Where the parameter *k* influences the unmarked trail's attractiveness and it is always greater than zero. If *k* is high, pheromone is high. The factor, *n* affects the linearity of decision. If n is higher, then the decision of choosing the best route also increases [55].

The same experiment when repeated with dissimilar path lengths as shown in Fig. The path B was made r times longer than the path A. Thus the effect called 'differential length effect' will affect the individual ant choosing the shortest path. Thus the ant choosing the shorter path initially will return to its nest quicker. Thus the deposit of pheromone in the shorter path is quicker which will make all the ants gradually to choose the shorter path than the similar binary bridge.

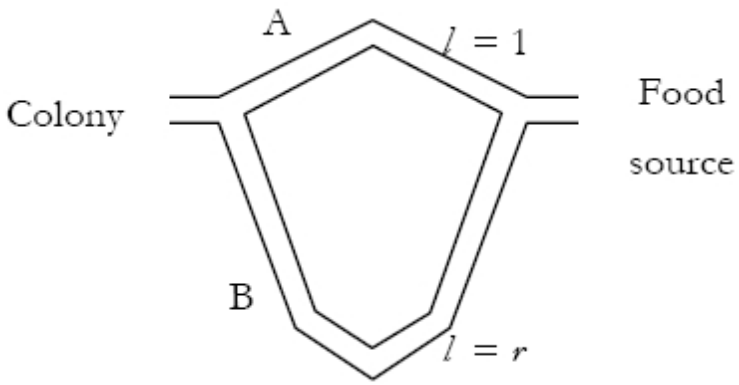


Figure 8: Binary bridge with unequal path length

It is great that only when ants search in colonies they find the shorter path with the help of indirect communication called pheromone.

The ant colony algorithm uses artificial ant rather than mimicking the exact real ant [55]. The artificial ant also builds the solution as the normal ant does. The unique feature of this algorithm is co-operation. There are some differences in the artificial ants than that of the real ants.

4.1 CHARACTERISITICS OF ARTIFICIAL ANTS:

The characteristic of the artificial ants are

1. Cooperation between individuals
2. Artificial pheromone for local communication
3. Sequences of moves to find short route
4. A decision part to decide the route with the help of local information

4.2 SIMILARITIES OF ARTICIAL ANTS AND REAL ANTS

Cooperation between individuals:

Similar to real ants, artificial ants are also in a population of colony. Each artificial ant will find a solution as a real ant does and cooperates with other ants in finding the best solution. The concurrent communication between the ants helps to find the best solution [55].

Pheromone trail:

As the real ants do modify their routes gradually because of some factors in the environment, the artificial ants too have their own numerical factors. The real ants deposit pheromone, a chemical substance in their path they visit. The next ants follow this pheromone. In similar way artificial ants store numeric information. The ant's current information and the performance are considered by this information and this numeric information can be read and modified by the next ants visiting the state. Thus this numeric information is only used as communication among the ants [55]. This stigmergetic information helps in using the collective knowledge. Even in ACO, the factor of evaporation is also considered. A subroutine would change the numerical information with respect to change in time. If no pheromone is deposited for a short time, then the strength of deposition of the pheromone decreases. Thus losing the pheromone information will make the new ant not to consider the past and take up a new route [55].

Sequence of moves:

The real ants and artificial ants have an objective of finding the shortest path from its origin to the destination. In reality the real ants will not jump. They just walk to next destination. Thus the shortest path is found step by step and not at once. The artificial ants also move step by step towards the destination.

Decision part:

For both the artificial and real ants a decision policy which is totally based on probability is needed to go from one step to other. Both the real and artificial ants will not predict the future. With respect to decision part the artificial ants have some specific characteristics:

1. Artificial ants are based on the computer which is a discrete world. Thus the steps of transition will be in discrete intervals.
2. They will have a memory to store the past actions
3. The amount of pheromone deposited by the artificial ant will determine the solution's quality.
4. In artificial ant the time interval in which the pheromone laid is dependent on the nature of the problem and is not similar to that of the real ant.

Thus to improve the efficiency of the solution, some abilities like look ahead [56], backtracking [57] and local optimization [58] could be used.

4.3 ACO META HEURISTICS:

In the Ant colony algorithm, a definite size of non-real artificial ants which possesses the quality as described above will search for a best solution to the considered problem. All the artificial ants will develop its own solution according to the requirement of problem. The artificial ant will store the information from the environment and the problem requirements according to the optimization objective. This is the stochastic information that is used to influence the other ants [55]. Thus by the use of co-operative behavior and stigmergetic communication, the solution is improved step by step constructively.

Each and every solution is constructed with the objective as minimal cost function. But the initial solution generated by the ants are always poor and of lesser quality. But the best quality solution will be developed step by step by the co-operation between the ants [55]. The ants move from one state to next neighborhood state which is completely based on the stochastic information stored and problem dependent.

The stochastic information is comprised of two things

1. The earlier history of the ant
2. Pheromone trail information and the heuristic information.

The information which is stored in the memory of the ant decides the quality of the solution formed. In multi objective combinatorial problem, some solutions are generated which makes it practically impossible to be applied [57]. So this memory functions as an important factor to determine the feasible solution as well.

The information from one ant to other ant is comprised of heuristic information that is specific to the problem, and the knowledge of pheromone trails deposited by all the ants in the search process. The quality and quantity of the pheromone released by each ant after the solution depends on the problem and the implementation design. The time at which the pheromone released can be of three types. Ants release the pheromone at the time building the solution (local update) or after the whole solution is built (global update) or both.

The speeding up of algorithm in framing the solution is more important. If a move is chosen by more ants, then pheromone deposited in that path is higher which will influence the next ant [57]. Quality of solution depends on the quantity of the pheromone. Ant decides on the next move based on the probabilistic table which directs the ant to the next appropriate state.

The design of algorithm with the help of stochastic component and pheromone mechanism will avoid all the ants getting into a similar choice. With the help of these two things all the solution is made to explore by the ants. The ant's decision policy could be enhanced with the help of minor add-ons to the algorithm like local optimization and backtracking.

In real ants, once a solution or path is found by the ant, it again returns to form the solution again. But the artificial ants, after giving a solution dies and is deleted. Ants' generation and the evaporation of the pheromone are also to be considered. Sometimes few more components like global perspective and daemon actions can also be added. The daemon action would bring a change in the stochastic information stored in the memory and would still more be helpful to find best solutions [58].

4.4 GLOBAL PROCEDURE OF ANTCOLONYMETAHEURISTIC [59]

Procedure ACO()

While (Termination condition is not satisfied)

Schedule activities

Ants generation and activity();

Pheromone evaporation();

Optional daemons action();

End scheduling of activities

End While

End procedure

Ant generation and activity

While (available resources)

Schedule the creation of new ant

New ant();

End while

End procedure

Procedure new active ant()

Initialize ant()

M= update ant memory()

While (current state ≠ target state)

A = read local ant routing table();

P= compute the transition probability (A, M, Ω);

Next state = apply ant decision policy (P, Ω)

Move to next state(next state);

If (online pheromone update)

Deposit pheromone on visited path();

Update ant routing table();

M= update internal state();

End while

If (delayed pheromone update)

For each visited path

Deposit pheromone on visited path()

Update ant routingtable()

End for each

Die();

End procedure

notations:

− *A*: set of routing tables
− *P:* set of probabilities
− *M*: memory of the ants
− Ω: set of constraints

4.5 CHOICES IN APPLICATION OFACO:

To apply ACO algorithm to a specific problem various wise choices must be made to get best results [59]. Firstly the pheromone trails must be decided and the heuristics that are required to depend on the environmental conditions must be decided. Next, the improvements of ACO like usage of backtracking or local search and number of ants. All these will have a great impact on the efficiency of the algorithm and its results.

Pheromone trails:

The important factor to be considered when applying ACO is the meaning of pheromone trail. In mTSP τ_{ij}, could refer to the desirability of visiting the city j from i. Here it is designed based on the objective of problem [59]. If the objective is to minimize the weighted tardiness, τ_{ij} represents the desirability of putting the job *i* in j^{th} position. But if the objective is to minimize the cost, then it represents the desirability a job *j* after job *i.* If definition of pheromone is bad then the algorithm will lead to poor performance.

Balance between exploration and exploitation:

The balance between the exploration and exploitation is equally important as pheromone trails. Exploration helps in exploring all the possible states of thee solution and this increases the chance of finding a very best solution. Pheromone trails tempt a probability distribution in the search space thus helping in exploration. After exploring and converging to a solution stagnation occurs. It is a point where all ants go through same tour.Exploitation of ants is done by updating pheromone [60]. Depending on the method of update, the quality of solution could be made stronger. A problem to be considered without heuristic information after running the algorithm for little time, the quantity of pheromone in each path differs. Thus the sampling focuses on a single space. Thus when the time increases the exploration decreases. One problem is stagnation as discussed earlier. This stagnation if occurs after very little time, it makes the algorithm to avoid searching the space where best solution would be available. To avoid this problem of stagnation different methods have been developed. In MMAS [60] developed algorithm with limits to pheromone trail. Method to reinitialize the pheromone trails would help the algorithm avoid stagnation.

Local search:

In mTSP, the ACO gives better solution when the local search is included in addition with ACO. After framing a solution a local search is applied and a locally optimized solution is got. This could be used as a information to update the pheromone trails. It has been proved that addition of local search gives better results [60] [61]. The major disadvantage of using local search technique in addition with the ACO is that it consumes lot of computational time.

Heuristic Information:

Heuristic information is problem dependant [61]. The knowledge about the problem must be included. The heuristic information in addition with pheromone is used to develop a solution.

Ant population:

It is considered to be better to use more than one ant. In some situations the number of states of the solution could be the number of ants. Mostly *m* ants are taken to construct *r* solutions, where *m*>1.

5.0 CODING IMPLEMENTATION:

5.1 PROBLEM DEFINITION:

The mTSP can be described with the help of a weighed and non-directed graph G (V, E). The Cities are represented by v_j, where j=1, 2….n. The tours between these cities can be represented by the letter e_i or the edges. $\omega(e_i)$ is the value which weighs the tour, that represent the cost of the tour. This cost may be distance, time etc. For *m* salesmen there must *m* closed loop tours: C_1, C_2, C_3...C_m and V_0. $V_0 \in C_i$, i =1, 2. . . m. Two objectives are considered as far mTSP is concern. (i) To minimize the individual cost of each salesman. (ii) To minimize the total cost of all salesmen. The problem can formulated as below.

$$\text{Min} \sum_{j=1}^{m} \sum_{ei \in Cj} \omega(ei) \quad (1)$$

$$\text{Min argmax}[\omega(Ci)] \quad (2)$$

$$\text{Subject to} \begin{cases} Vo \in Ci(V)(i = 1,2, \ldots . m) \\ \cup_{i=1}^{m} C(V) = G(V) \end{cases} \quad (3)$$

The above equation defines the problem. The Eq. (1) defines one of the objectives i.e., minimizing the total length of the route that all salesmen travel. Eq. (2) defines the other

objective minimizing the path length travelled by each salesman individually. Eq. (3) defines the depot system.

5.2 ANT COLONY ALGORITHM FOR mTSP:

A modified nearest neighbour algorithm is used to formulate the initial solution. A random city is selected by each salesman and then nearest cities is visited until all the cities are visited. This continues until the initial solution is generated. After the initial solution is being constructed then the operator uses the pheromone information and the heuristics. The heuristics includes the path length; the pheromone trails used are similar to that used in TSP. To improve solutions, next city is selected according to the pseudo-random-proportional rule.

$$\text{j=}\begin{cases}\arg\max_{k\in S}\left\{[\tau_{ik}]^{\alpha}.[\boldsymbol{\eta}_{ik}]^{\beta}\right\}, if\ q \le q_0 \\ J \qquad otherwise\end{cases} \tag{4}$$

where $\tau_{ik}(t)$ is the value of pheromone trail between city *i* and *k* at the time *t*. $\boldsymbol{\eta}_{ik}$ $(\boldsymbol{\eta}_{ik} = 1/\mathbf{d}_{ik})$ is the visibility of city *k* from the city *i*. α, β represents pheromone trail relative importance and the visibility respectively. S is the set of unvisited cities. *q* is randomly chosen within the unfirom probability in [0,1]. q_0 is a smaller parameter chosen for higher random selection. J is a random variable according to the probability defined as follows, $P_{ij}(t) = \begin{cases}\frac{[\tau_{ij}(t)]^{\alpha}[\boldsymbol{\eta}_{ij}]^{\beta}}{\sum_{k\epsilon S}[\tau_{ik}]^{\alpha}[\boldsymbol{\eta}_{ik}]^{\beta}}, if\ j\epsilon S \\ 0 \qquad otherwise\end{cases}$

(5)

The operator chooses the next city which is the fittest with probability q_0 $(0 \le q_0 \le 1)$ and next city is selected with the probability as mentioned in Ed. 5 until all cities are visited.

5.2.1 Pheromone Update:

The pheromone update rule is given as below,

$$\tau = \frac{1}{1-\rho}\frac{1}{f(S^{gb})} \tag{6}$$

Where ρ is the pheromone trail persistence, f(S^{gb}) is the tour length of globally best solution.

The pheromone is updated on the basis of the globally best solution. The pheromone is updated based on the following rule.

$$\tau_{ij}(t+1) = \rho\tau_{ij}(t) + \Delta\tau_{ij}^{gb} \quad (7)$$

$$\text{Where } \Delta\tau_{ij}^{gb} \begin{cases} \frac{1}{f(S^{gb})}, if\ (i,j) \epsilon\ S^{gb} \\ 0, \qquad otherwise \end{cases} \quad (8)$$

Eq. 8 decides that only those edges belonging to the globally best solution.

5.2.2 Local Search:

The local search is based on three universally known operators. This will help to get the better solution. These operators are 2*-opt operator, exchange operator and relocation operator. The local search is applied to the best solution. These heuristics will improve the solution. The relocation heuristic removes city from one route and it is re-inserted into other route. The exchange operator swaps the route simultaneously. Two different cities in two different routes are exchanged. These two operators are tried for all the possible combinations. The 2-opt* operator swaps the ends of two routes by conserving the orientation and maintaining the feasibility. When this local search occurs and if there is a reduction in length then the solution is updated. This local search increases the iteration time.

5.2.3 Implementation of ACO

The ACO cannot be applied as it can be used for a normal TSP. mTSP has multiple objective. So, few modifications are made to the Ant Colony Optimization to solve mTSP. In normal TSP a random solution would be chosen and then the optimization process starts. But here the initial solution is chosen by nearest neighbour algorithm.

5.2.4 Explanation of Program:

Pseudo code:

The programming has been done with the help of the pseudo code mentioned below

START

Initializing the parameters and Pheromone

Initializing the ant memory

While terminal condition is not satisfied do

For each ant

For each salesman

Generate initial solution using nearest neighbor algorithm

End for

Initializing pheromone;

Construct solution according to the meta-heuristics;

Update the pheromone trails;

Apply Local Search;

End for

Pheromone Evaporation;

Global pheromone update;

Update the ant memory;

End while

STOP

6.0MATLAB CODE EXPLANATION:

6.1. ANT COLONY ALGORITHM

The Pseudo code above is the basic code. The complete code was developed with the help of the above pseudo code. The complete code was developed with a good programming knowledge, basic functions and structural array.

Firstly the input file is read from the program. The input file has the number of salesman, and the 2D details of the cities located. The input file parameters are read and stored. Now the initial parameters are set.

Initial Parameters:

Maximum iteration, number of ants, alpha, beta, and rho are the initial parameters defined.

Alpha - Relative importance of the pheromone trail

Beta - visibility of the pheromone trail

Rho - Pheromone trail persistence.

Distance matrix is computed in a separate function and stored.

AS is the main function that applies Ant colony algorithm. For the sake of easiness in programming a separate function called 'ASOption' was done. All the initial values were initialised under this function so that it could be denoted as a global variable under different functions. Also, a local search decision factor has been included. If applylocalsearch is 0, the local search process will not apply to the program. Other initial parameters required by the ant system like, Q, C, and lambda are initialised under this function.

Next function used is 'Problem'. This initializes the tau matrix which is the ant memory to store pheromone trails. Again for the easiness of programming, a structure called Problem is initialized which stores the Nodes, Distance and MatrixTau in structural array format.

Iteration starts for solution improvement. The initial solution is developed within two functions; AntSystem and starting points. The initial solution is random with some known heuristics. The firs salesman chooses a random point. Then next salesman chooses the point farthest from salesman 1. Then it continues for all salesmen. Then nearest neighbour algorithm is used to build up and complete the solution. For each ant the problem continues to build solution with function 'nextnode'.

Each salesman should reach the initial city visited. To do this task a function called "Close Tours" is used. The tours are recorded with the function 'tours'. Then the local search is applied if needed.

Pheromone is then updated. Refresh pheromone function helps saving the pheromone trail as a basic ant does. The pheromone is updated for all ants. The probability determination of next tour depends on this pheromone information. 'Sumtau' is the pheromone update variable and Problem. tau is the variable for pheromone decay.

Now after refreshing, the best solution is taken into account among all the ants. The tours are recorded. For the current best solution, the graph is drawn by the function 'DrawCity'.

Local Search:

The routes are stored in the array format. If the initial solution of the 2 salesman for 9 city problem is as follows.

Initial solution:

4	**6**	**3**	**9**

1	**2**	**5**	**7**	**8**

After relocation operation: (single step)

6	**4**	**3**	**9**

5	**2**	**1**	**7**	**8**

After Exchange operator: (single step)

5	**2**	**3**	**9**

↕ ↕

6	**4**	**1**	**7**	**8**

After 2* Opt operator: (single step)

2	**5**	**9**	**3**

4	**6**	**1**	**8**	**7**

Thus after all the operations, if the solution is found to be optimal it will updated.

6.2 GENETIC ALGORITHM**:**

GA is applied to the mTSP here. The type of chromosome representation is the two-part chromosome representation as discussed earlier.

The variable xy has the number of cities that are in the problem in 2 D format. The parameters initialized are

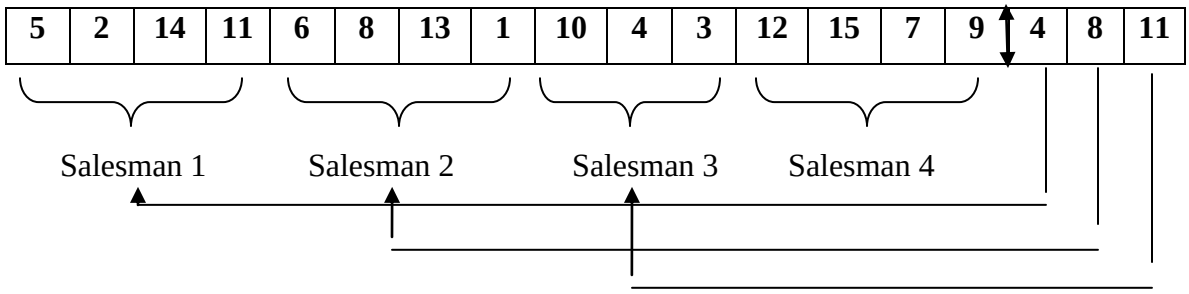
1. Number of salesman
2. Minimum tour of each salesman
3. Population size (Must be in multiples of 8)
4. Maximum number of iterations.

Now the distance Matrix is computed. The representation of chromosome is two-part chromosome. The genetic algorithm considered here is completely based on randomness. The cities are arranged in random in the variable popRoute. The popBreak has the random breaks. The number of breaks applied is (n-1), where n is the number of salesman. The 'routeInsertionPoint' is a variable that holds the place where to perform the operation on cities and breaks.

Now the chromosome is made of eight genes. The first gene is the best solution of the initial population generated. Next gene undergoes a flip operation in cities. Cities are flipped. For example in a

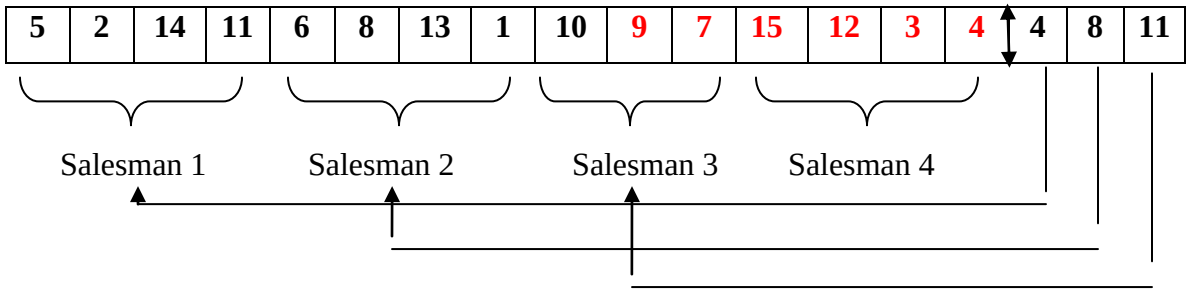
1st gene

This is the first gene and the least total distance of the solution generated in random



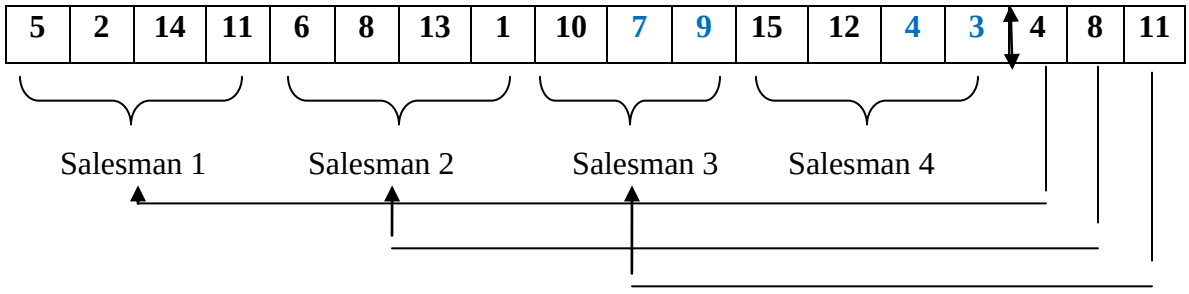
2nd gene after flip operation:

Flip operation is performed where the distance gets minimum. A group of cities is flipped as highlighted below.



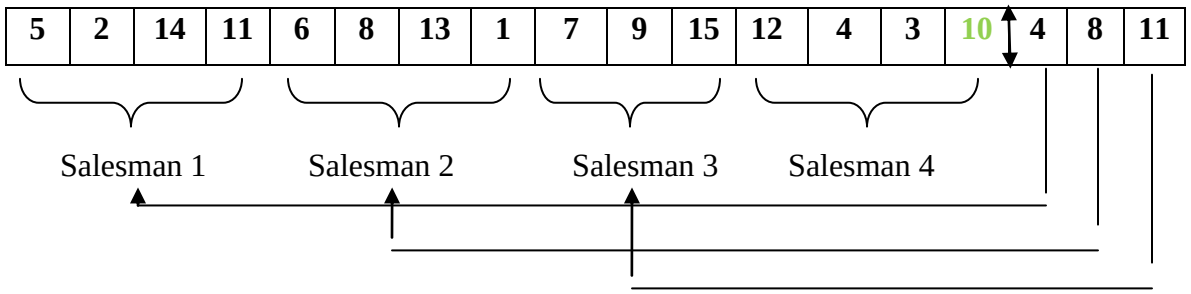
3rd gene after swap operation:

The Swap operation is performed in the cities where the best solution is possible. Breaks are untouched.



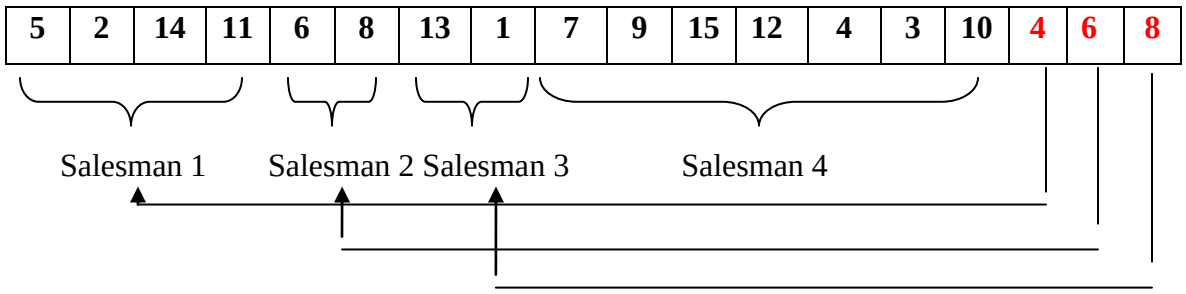
4th gene after slide operation:

One of the cities slides into some position. And the other cities are adjusted accordingly. The changes are highlighted.



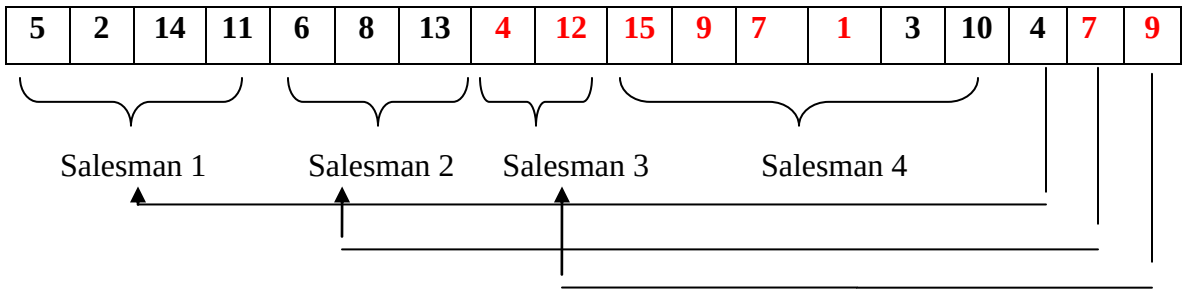
5th gene after modifying break:

The breaks are now determined in random and highlighted below.



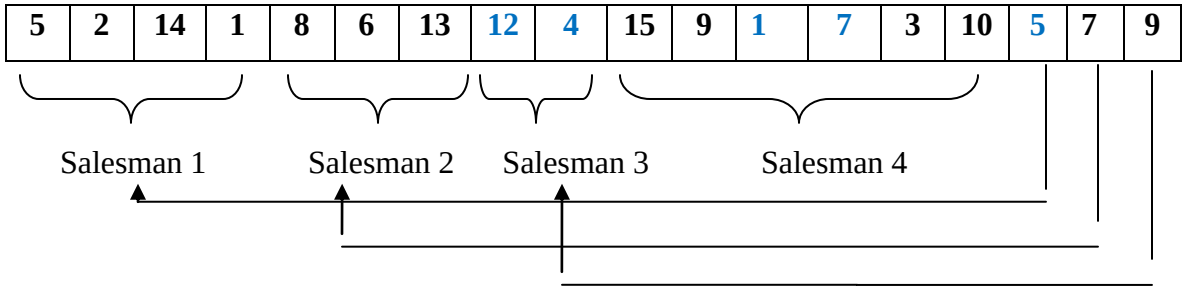
6th gene after flip operation and modifying break:

Both the break modification and flip of the cities occurs.

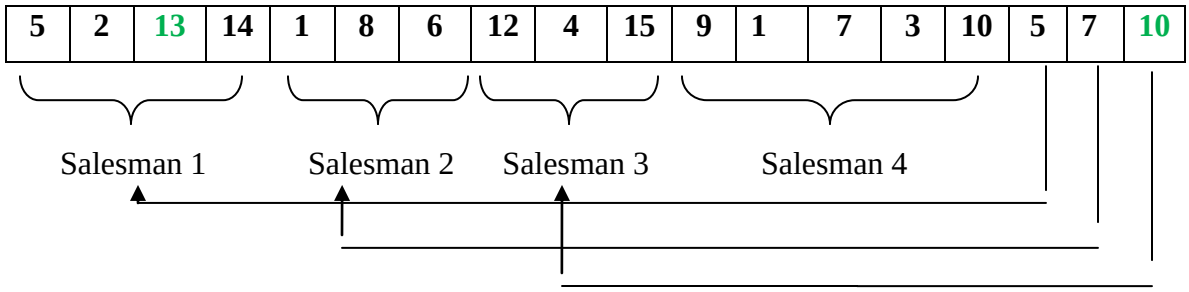


7th gene after swap operation and modifying break:

The swap operation is performed in addition with the break modification.



8th gene after slide operation and modifying break:



The best solution is then saved and the iteration works. Thus the best solution is kept as the first gene. The next iteration occurs until the maximum iteration is reached or the best solution is found.

7.0 RESULTS:

The effectiveness of each algorithm was compared separately and also together.

TSP:

i. 20 cities

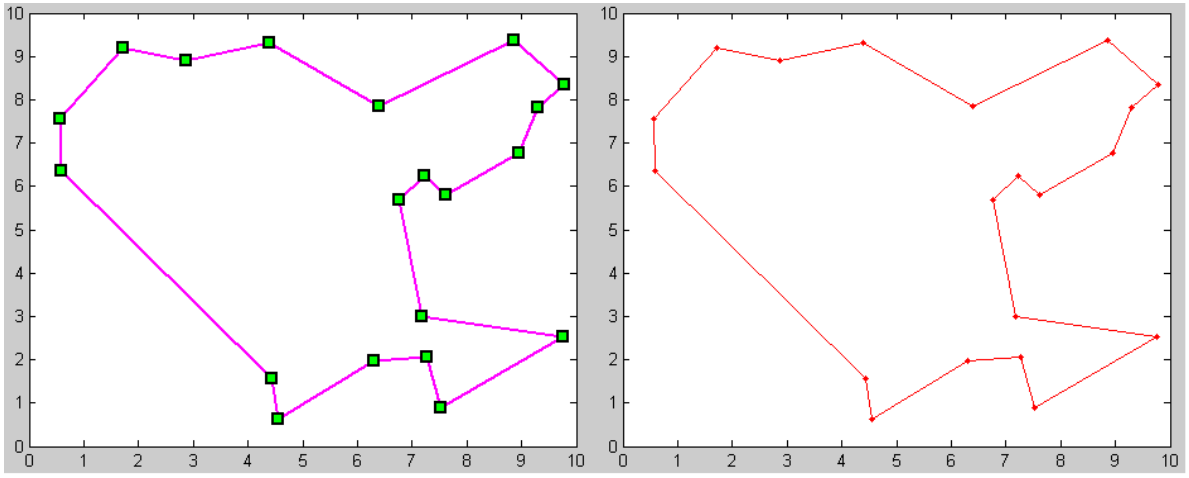


Figure 9: Final result of Ant colony algorithm and Genetic algorithm for 20 cities. Ant Colony Result (iteration: 628, least length: 37). Genetic Algorithm Result (iteration: 93 length: 37)

ii. 30 cities:

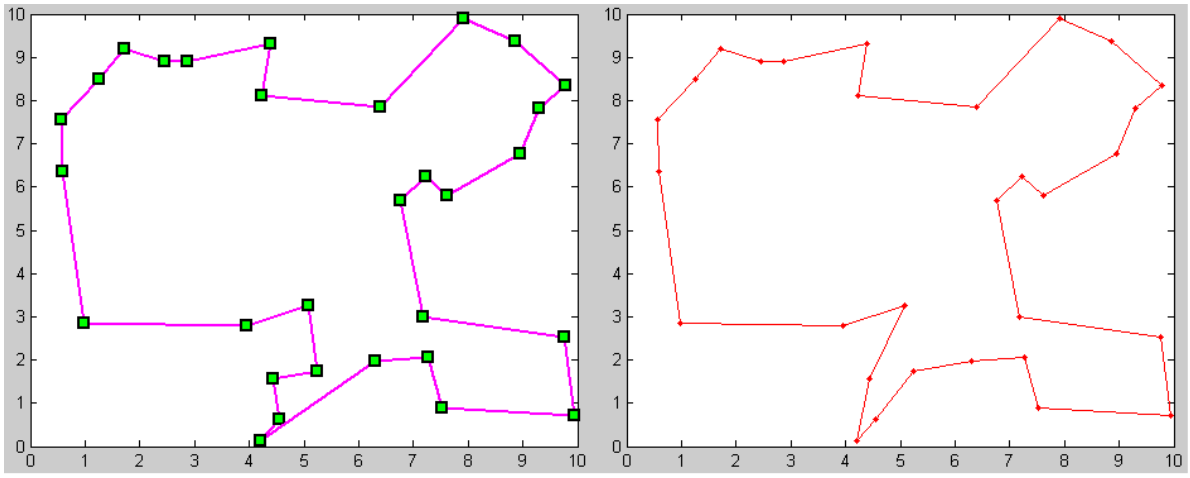


Figure 10: Final result of Ant colony algorithm and Genetic algorithm for 30 cities. Ant Colony Result (iteration: 85, least length: 45.29). Genetic Algorithm Result (iteration: 3607 length: 44.86)

iii. 50 Cities:

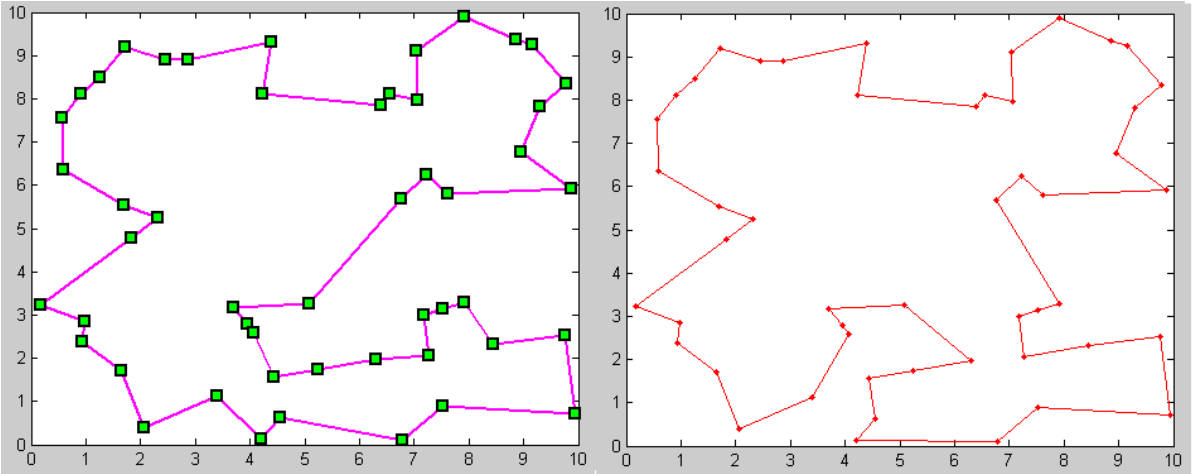


Figure 11: Final result of Ant colony algorithm and Genetic algorithm for 30 cities. Ant Colony Result (iteration: 23, least length: 55.46). Genetic Algorithm Result (iteration: 2883 length: 54.75)

iv. 75 cities:

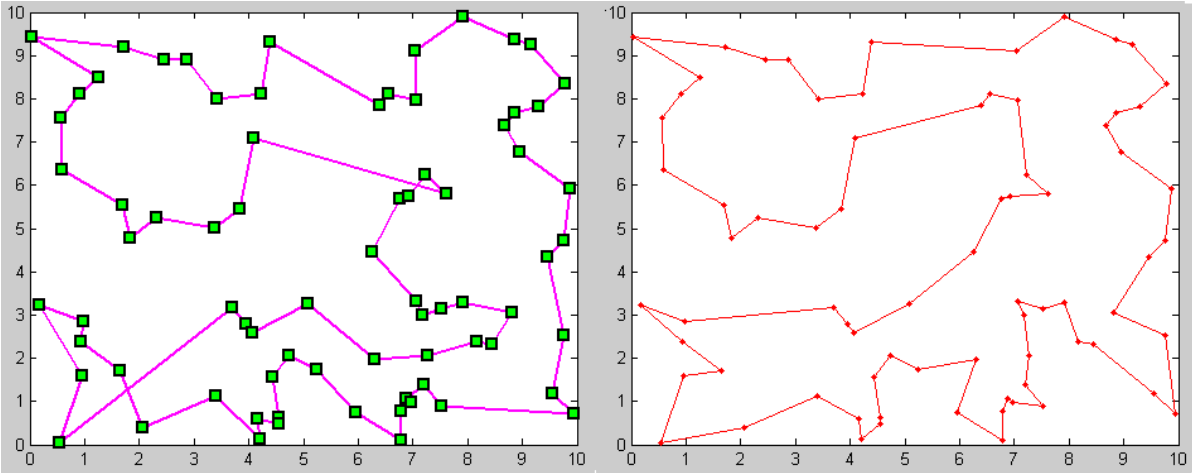
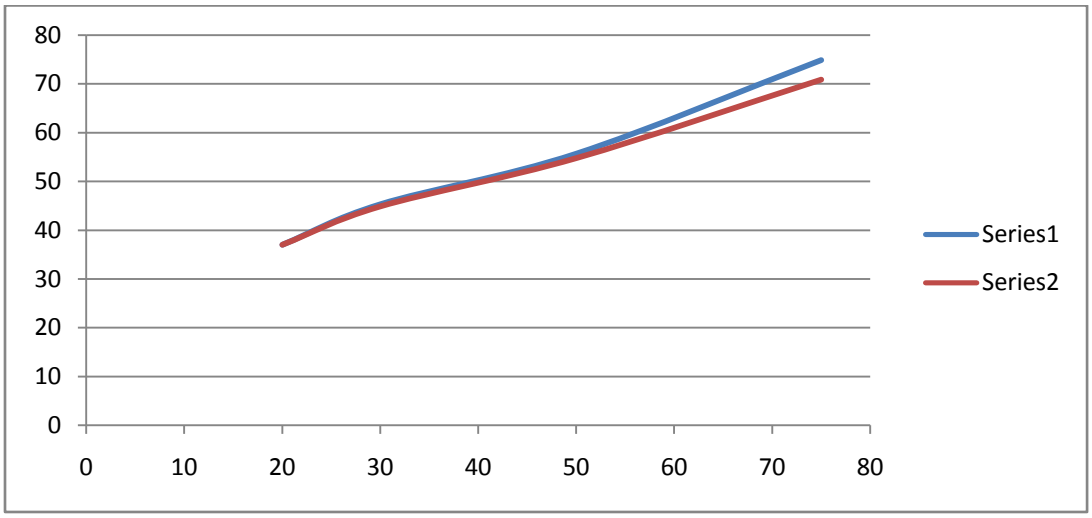


Figure 12: Final result of Ant colony algorithm and Genetic algorithm for 30 cities. Ant Colony Result (iteration: 76, least length: 74.83). Genetic Algorithm Result (iteration: 4474length: 70.83)



Graph 1: Pareto plot between the no. of cities VS the best solution of GA and best solution of ACO for TSP

It could be clearly seen in a single objective problem, as the complexity increases the GA performs well when compared to ACO. Also the Ant Colony algorithm performs the iteration at a slower speed.

MTSP

(i) 20 Cities

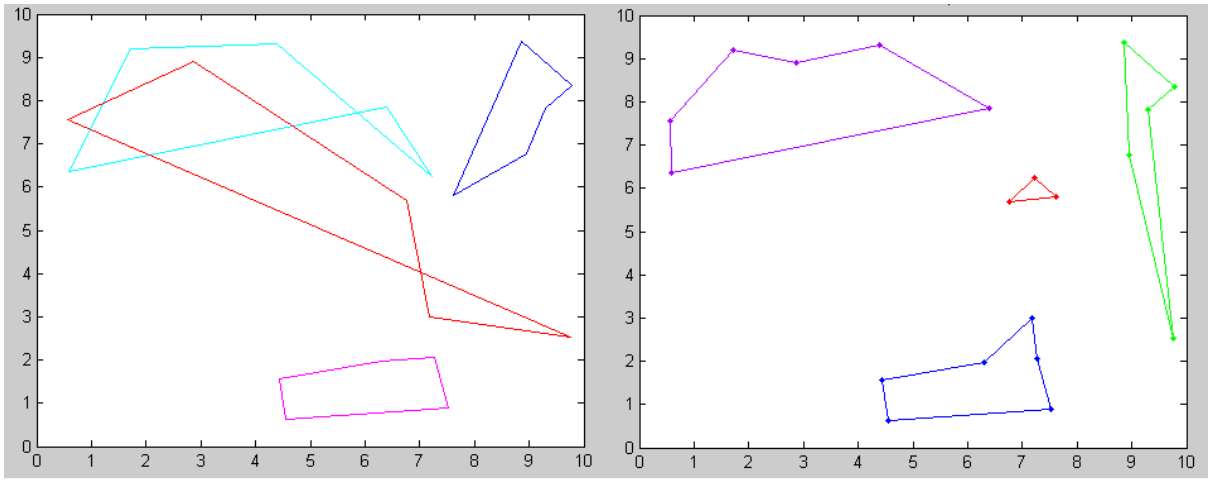


Figure 13: Final result of Ant colony algorithm and Genetic algorithm for 20 cities. Ant Colony Result (iteration : 582, Least length: 50.8). Genetic Algorithm Result (iteration: 779 length: 41.18)

(ii) 30 cities

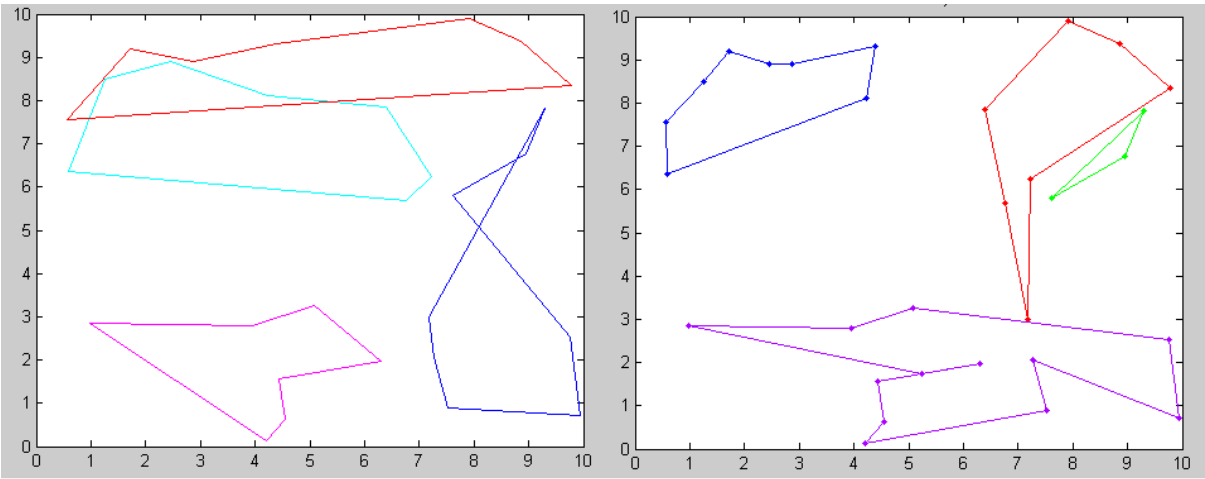


Figure 14: Final result of Ant colony algorithm and Genetic algorithm for 30 cities. Ant Colony Result (iteration : 372 Least length: 67.41). Genetic Algorithm Result (iteration: 987 length: 60.40)

(iii) 50 cities

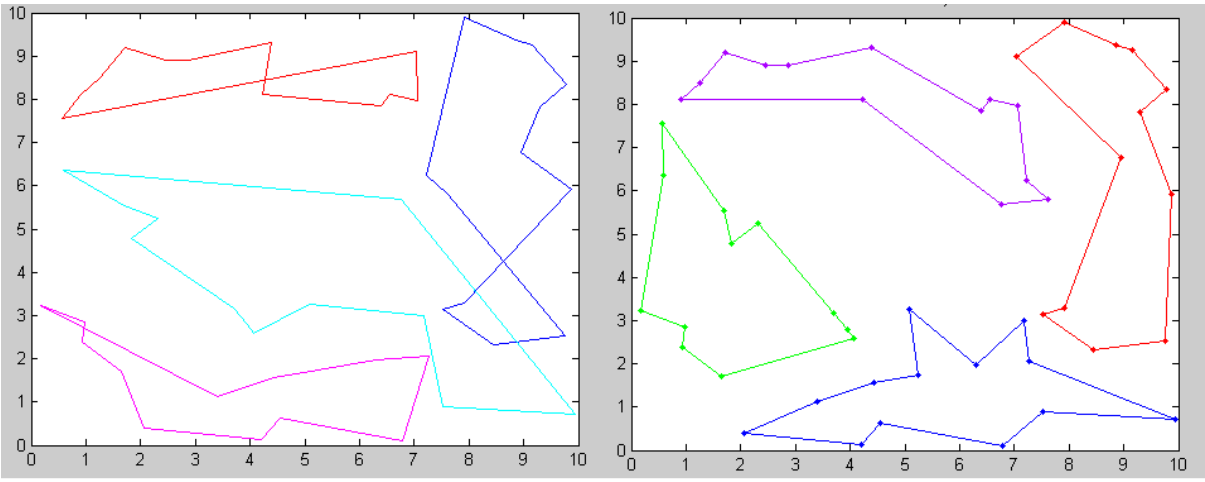


Figure 15: Final result of Ant colony algorithm and Genetic algorithm for 50 cities. Ant Colony Result (iteration : 557, Least length: 80.3). Genetic Algorithm Result (iteration: 4165 length: 73.8)

(iv) 75 cities

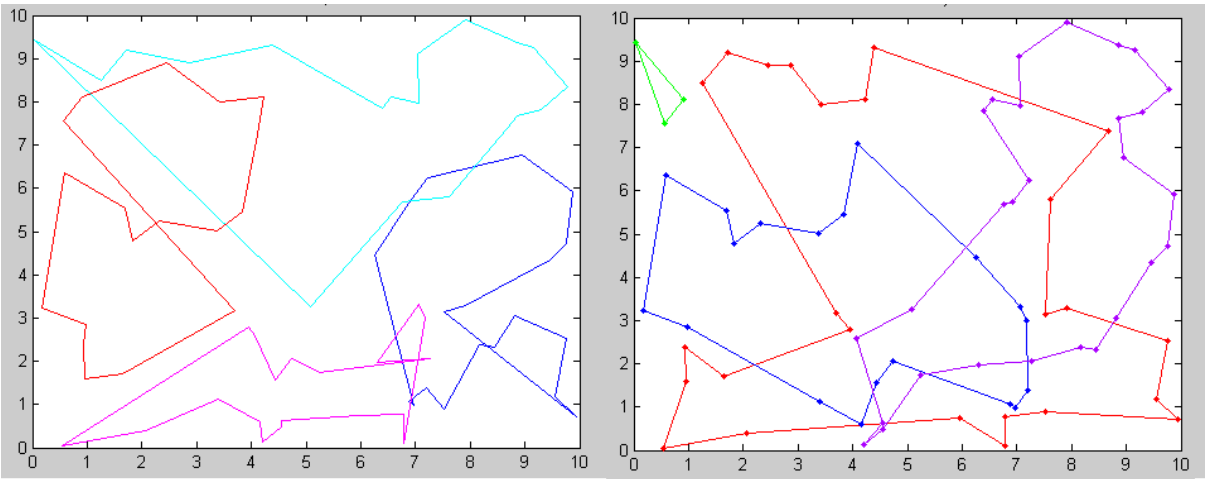
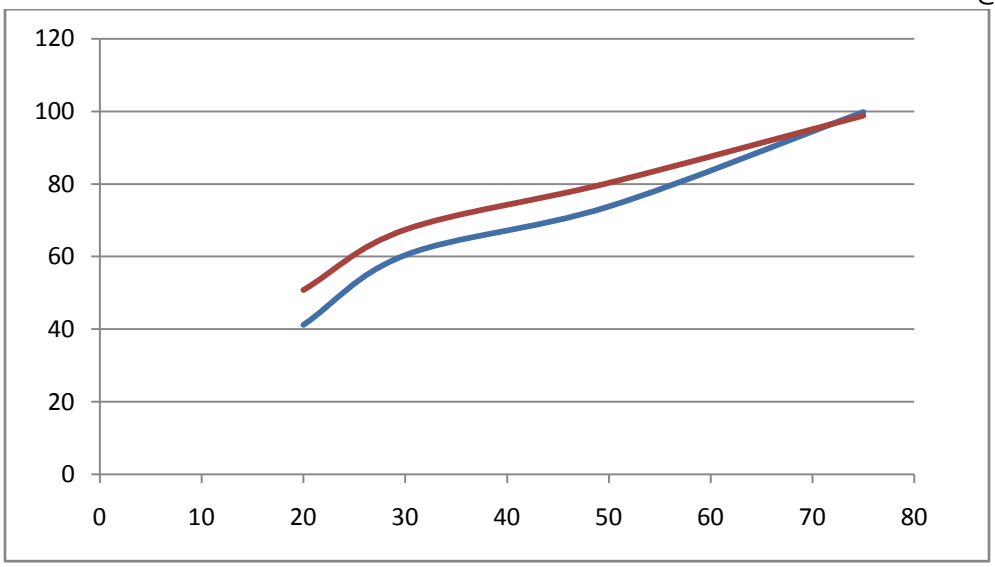


Figure 16: Final result of Ant colony algorithm and Genetic algorithm for 20 cities. Ant Colony Result (iteration : 117, Least length: 98.8). Genetic Algorithm Result (iteration: 14787 length: 99.8)

It can be clearly seen that in all other figures except fig 16, the best solution of ACO is more than the GA. But in Fig 12, with 75 cities, the best solution is lesser in ACO than in GA. So if complexity increases, ACO will be able to give a better solution than GA.



Graph 2: Pareto plot between the no. of cities VS the best solution of GA and best solution of ACO for mTSP

8.0. DISCUSSION:

The main objective of this dissertation was to develop an ant colony algorithm code and compare its result with that of Genetic algorithm code that was developed to solve the same problem. However it was found that GA was more effective in solving the mTSP for smaller number of cities. The development of algorithm was done with high amount of self learning. Great support was given by Prof. Antony Jones to develop the Genetic algorithm code. The mTSP was not researched by many, which made me to put immense work on it. Initially the program, which was used to solve TSP with Ant Colony Algorithm and Genetic Algorithm was studied and understood. After that ACO and GA code for multi-objective TSP was developed.

There are some factors which affects both the algorithm. For ACO, development of local search process took a heavy work and time. It was hard initially to visualise how the process works. Then structural array concept was used in the algorithm to solve the problem. The other factors like, co-efficient of evaporation, alpha and beta, had impact on the final result. If proper care and further research is put up, better Ant colony algorithm could be developed which is very fast. The one developed took a lot of time to complete the iteration process.

Genetic algorithm was quite easier than ACO to work with. More and more research works were available for GA solving of TSP which helped me to develop the code. Research on both these algorithms helped me to understand the application areas of all these algorithms. There is more and more application with regarding mTSP which has lesser research.

Immense research is needed to quicken the Ant colony algorithm process. If the ACO needs to be applied to a practical application, it will demand a higher processor for the algorithm to work at a normal speed. The time of convergence is high when compared to GA, but the real fact is ACO gives better prospective solution with higher computing complexity.

9.0 CONCLUSION AND FUTURE RESEARCH:

Thus from the above results it can be found that, ACO best works for more number of cities. Since the GA does not use any heuristic information to create a solution, it also takes a lot of iterations to solve the mTSP with higher number of cities. Ant colony algorithm uses the heuristic information and local search techniques to frame a solution and to converge towards the best solution. The only problem to be eradicated with respect to ant colony algorithm is to hasten its iteration speed. The iteration time is lot more compared to GA.

10.0 REFERENCES:

[1]. Dorigo, M. (1992) *Optimization, learning and natural algorithms.* A Thesis Submitted in partial fulfilment of the Requirements of Dipartimento di Elettronica, Politecnico di Milano, Italy for the Degree of Doctor of Philosophy. Stoke-on-Trent: Staffordshire University.

[2]. Oberlin, P.; Rathinam, S.; Darbha, S. (2009) "A transformation for a Heterogeneous, Multiple Depot, Multiple Traveling Salesman Problem," *American Control Conference, 2009. 09,* pp.1292-1297.

[3]. Donald, D. (2010) *Traveling Salesman Problem, Theory andApplications*. 9th Ed. InTech.

[4]. Yadlapalli, S.; Malik, W.A.; Darbha, S.; Pachter, M.; , "A Lagrangian-Based Algorithm for a Multiple Depot, Multiple Travelling Salesmen Problem," *American Control Conference, 2007. ACC '07* , vol., no., pp.4027-4032, 9-13 July 2007

[5]. Geetha, R.R.; Bouvanasilan, N.; Seenuvasan, V.; , "A perspective view on Travelling Salesman Problem using genetic algorithm," *Nature & Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on* , vol., no., pp.356-361, 9-11 Dec. 2009

[6]. Lishan Kang; Aimin Zhou; McKay, B.; Yan Li; Zhuo Kang; , "Benchmarking algorithms for dynamic travelling salesman problems," *Evolutionary Computation, 2004. CEC2004. Congress on* , vol.2, no., pp. 1286- 1292 Vol.2, 19-23 June 2004

[7].Dorigo, M. and Di Caro, G. (1999) The ant colony optimization meta-heuristic. *New Ideas in Optimization*, McGraw-Hill, London, UK, pp.11–32.

[8]. Potvin, J. Lapalme, G. Rousseau, J. (1989) A generalized k-opt exchange procedure for the mTSP. *INFOR* [online]. 21, pp. 474–481.

[9]. Russell, R.A. (1977) An effective heuristic for the m-tour traveling salesman problem with some side conditions. *Operations Research* [online]. 25(3), pp. 517–524.

[10]. Svestka, J.A. &Huckfeldt, V.E. (1973) Computational experience with an m-salesman traveling salesman algorithm. *Management Science* [online]. 19 (7), pp. 790–799.

[11] Lenstra, J.K. and RinnooyKan, A.H.G. (1975). Some simple applications of the traveling salesman problem.*Operational Research Quarterly*, [online] 26, pp. 717–733.

[12] Zhang, T.; Gruver, W.A. and Smith, M.H. (1999). Team scheduling by genetic search.*Proceedings of the second international conference on intelligent processing and manufacturing of materials* [online] 2 pp. 839–844.

[13] Brummit, B. andStentz, A. (1996). Dynamic mission planning for multiple mobile robots.*Proceedings of the IEEE international conference on robotics and* automation [online]

[14] Brummit, B. andStentz, A. (1998). GRAMMPS: a generalized mission planner for multiple mobile robots. *Proceedings of the IEEE international conference on robotics and automation*[online]

[15] Saleh, H.A. andChelouah, R. (2004). The design of the global navigation satellite system surveying networks using genetic algorithms.*Engineering Applications of Artificial Intelligence* [online] 17, pp. 111–122.

[16] Gorenstein, S. (1970) Printing press scheduling for multi-edition periodicals. *Management Science* [online] 16 (6), pp. 373–383.

[17] Carter, A.E. and Ragsdale, C.T. (2002) Scheduling pre-printed newspaper advertising inserts using genetic algorithms. *Omega*[online].30, pp. 415–421.

[18]Tang, L.; Liu, J.; Rong, A. and Yang, Z. (2000) A multiple traveling salesman problem model for hot rolling scheduling in ShangaiBaoshan Iron & Steel Complex. *European Journal of Operational Research* [online]. 124, pp. 267–282.

[19] Gilbert, K.C. and Hofstra, R.B. (1992) A new multiperiod multiple traveling salesman problem with heuristic and application to a scheduling problem. *Decision Sciences*[online]. 23, pp.250–259.

[20] Angel, R.D.; Caudle, W.L.; Noonan, R. andWhinston, A. (1972) Computer assisted school bus scheduling. *Management Science* [online].18, pp.279–288.

[21]Laporte, G. &Nobert, Y. (1980) A cutting planes algorithm for the m-salesmen problem.*Journal of the Operational Research Society* [online].31, pp.1017–1023.

[22]Gavish, B. &Srikanth, K. (1986) An optimal solution method for large-scale multiple traveling salesman problems. *Operations Research* [online]. 34 (5), pp. 698–717.

[23] Gromicho, J.;Paixão, J. &Branco, I. (1992) Exact solution of multiple traveling salesman problems. *Combinatorial optimization. NATO ASI Series* [online]. F82, pp. 291–292.

[24]Arora, S. (1998) Polynomial Time Approximation Schemes for Euclidian Traveling Salesman and Other Geometric Problems. *Journal of ACM* [online]. 45(5), pp. 753-782.

[25] Johnson D.S. &McGeoch L.A. (1995) The Traveling Salesman Problem: *A Case Study in Local Optimization* [online].

[26] Applegate, D.L.; Bixby, R.E.; Chv´atal, V.; Cook, W.J.; Espinoza, D.G.; Goycoolea, M. &Helsgaun, K. (2009) Certification of an optimal TSP tour through 85900 cities. *Operations Reearch Letters* [online]. 37 (1), pp.11–15.

[27] Applegate, D.L.; Bixby, R.E.; Chv´atal, V.; Cook, W.J.; Espinoza, D.G.; Goycoolea, M. &Helsgaun, K. (2009) Certification of an optimal TSP tour through 85900 cities. *Operations Reearch Letters* [online]. 37 (1), pp.11–15.

[28] Johnson D.S. &McGeoch L.A. (1995) The Traveling Salesman Problem: *A Case Study in Local Optimization* [online].

[29] Piwonska, A.; Seredynski, F.; , "A Genetic Algorithm with a Penalty Function in the Selective Travelling Salesman Problem on a Road Network," *Parallel and Distributed Processing Workshops and Phd Forum (IPDPSW), 2011 IEEE International Symposium on* , vol., no., pp.381-387, 16-20 May 2011

[30]Dorigo, M. & Gambardella, L.M. (1996).*Ant Colonies for the Traveling Salesman Problem*.A Thesis Submitted in partial fulfilment of the Requirements of University Libre de Bruxelles, Belgium for the Degree of Doctor of Philosophy University Libre de Bruxelles, Belgium.

[31] Potvin, J.; Lapalme, G. & Rousseau, J. (1989) A generalized k-opt exchange procedure for the MTSP. *Information Systems*[online].27(4), pp. 474–481.

[32] Potvin, J.; Lapalme, G. and Rousseau, J. (1989) A generalized k-optexchange procedure for the MTSP. *INFOR*[online]. 27(4), pp. 474–481.

[33] Wacholde, E.; Han, J. and Mann, R.C. (1989) A neural network algorithmfor the multiple traveling salesmen problem. *Biology inCybernetics* [online].61, pp. 11–19.

[34] Hsu, C.; Tsai, M. and Chen, W. (1991) A study of feature-mappedapproach to the multiple travelling salesmen problem.*IEEE International Symposium on Circuits and Systems* [online]. 3, pp. 1589–1592.

[35] Vakhutinsky, I.A. and Golden, L.B. (1994) Solving vehicle routing problemsusing elastic net. *Proceedings of the IEEE international conference on neural network* [online].8 (7) pp. 4535–4540.

[36]Modares, A.;Somhom, S. andEnkawa, T. (1999) A self-organizing neuralnetwork approach for multiple traveling salesman and vehiclerouting problems. *International Transactions in OperationalResearch*[online]. 6, pp. 591–606.

[37] Somhom, S.;Modares, A. and Enkawa, T. (1999) Competition-basedneural network for the multiple traveling salesmen problemwith minmax objective. *Computers and Operations Research*[online].26(4), pp. 395–407.

[38] Zhang, T.;Gruver,W.A. and Smith, M.H. (1999) Team scheduling by geneticsearch. *Proceedings of the second international conference onintelligent processing and manufacturing of materials* [online]. 2, pp. 839–844.

[39] Tang, L.; Liu, R.;Rong, A. and Yang, Z. (2000) A multiple traveling salesmanproblem model for hot rolling scheduling in ShangaiBaoshanIron & Steel Complex. *European Journal of OperationalResearch*[online].12(4), pp. 267–282.

[40] Yu, Z.;Jinhai, L.;Guochang, G.;Rubo, Z. andHaiyan, Y. (2002)Animplementation of evolutionary computation for path planningof cooperative mobile robots. *Proceedings of the fourth worldcongress on intelligent control and automation*[online]. 3, pp. 1798–1802.

[41] Ryan, J.L.; Bailey, T.G.; Moore, J.T. and Carlton, W.B. (1998) ReactiveTabu search in unmanned aerial reconnaissance simulations.*Proceedings of the 1998 winter simulation conference*[online]. 1, pp. 873–879.

[42] Paul, M. and Thompson, N. (1993) Cyclic transfer algorithmsfor mutivehiclerouting and scheduling problems. *Operations Research* [online]. 41 (5), pp. 935-946.

[43] Samerkae, S.;Abdolhamid, M. and Takao, E. (1999) Competition-based neural network for the multiple travelling salesmen problem with minmax objective.*Computers & Operations Research* [online]. 26, pp. 395-407.

[44] Sheldon, H.; Jacobson, A.; McLay, N.; Hall, A.;Darrall, H.; Diane, E. and Vaughan, B. (2006) Optimal search strategies usingsimultaneous generalized hill climbing algorithms.*Mathematical and Computer Modelling* [online].43, pp. 1061–1073.

[45] Ching Fang, L. (2000) A hybrid genetic algorithm for the open shop scheduling problem.*European Journal of Operational Research* [online].124,pp. 28-42.

[46] Goldberg, D.E. and Lingle, R.J. (1985) Alleles, loci, and the traveling salesman problem.*Proceedings of An International Conference on Genetic Algorithms* [online]. 7 (3), pp. 154-159.

[47] Malmborg, C. (1996) A genetic algorithm for service level based vehicle scheduling.*European Journal of Operational Research* [online]. 93(1), pp. 121-134.

[48] Park, Y.B. (2001) A hybrid genetic algorithm for the vehicle scheduling problem with due times and time deadlines. *International Journal of Productions Economics* [online]. 73(2), pp. 175-188.

[49] Holland, J.H. (1975). *Adaptation in Natural and Artificial Systems*. MIT Press, Cambridge.

[50] Jog, P.; Suh, J.Y. and Van Gucht, D. (1989). The effects of population size, heuristic crossover and local improvement on a genetic algorithm for the traveling salesman problem, *Proceedings of the Third International Conference on Genetic Algorithms*, *Los Altos, CA* [online]. pp. 110-115.

[51] Goldberg, D.E. and Lingle, R.J. (1985) Alleles, loci, and the traveling salesman problem.*Proceedings of An International Conference on Genetic Algorithms* [online]. 7 (3), pp. 154-159.

[52] Potvin, J. (1996) Genetic algorithms for the traveling salesman problems. *Annals of Operations Research* [online]. 63 (1), pp. 330-370.

[53] Ragsdale, C.T. (2001) *Spreadsheet Modeling and Decision Analysis*.3rd edition, South-Western College Publishing, Cincinnati, Ohio.

[54] Pan Junjie; Wang DingWei; , "An Ant Colony Optimization Algorithm for Multiple Travelling Salesman Problem," *Innovative Computing, Information and Control, 2006. ICICIC '06. First International Conference on* , vol.1, no., pp.210-213, Aug. 30 2006-Sept. 1 2006

[55]Deneubourg, J. L.; Aron, S.; Goss, S. andPasteels, J. M. (1990) The self organizingexploratory pattern of the Argentine ant.*Journal of Insect Behaviour*[online]. 3, pp. 159.

[56] Michel,R. and Middendorf, M. (1998)An island model based ant system with lookaheadfor the shortest supersequence problem. *Proceedings of PPSN-V, Fifth International Conference on ParallelProblem Solving from Nature*[online]. 6 (4), pp. 692–701.

[57]Di Caro, G. and Dorigo, M. (1997) *AntNet: A mobile agents approach to adaptive routing.*Technical Report IRIDIA, UniversitéLibre de Bruxelles.

[58] Dorigo, L. and Gambardella, M. (1997)Ant Colony System: A cooperative learningapproach to the travelling salesman problem. *IEEE Transactions on EvolutionaryComputation* [online]. 1 (1), pp. 53–66.

[59] Gambardella L. M. andDorigo, M. (1995) Ant-Q: A reinforcement learning approach to the travelling salesman problem.*Proceedings of the Twelfth International Conferenceon Machine Learning (ML-95)* [online].pp. 252–260.

[60] Pan Junjie; Wang DingWei; , "An Ant Colony Optimization Algorithm for Multiple Travelling Salesman Problem," *Innovative Computing, Information and Control, 2006. ICICIC '06. First International Conference on* , vol.1, no., pp.210-213, Aug. 30 2006-Sept. 1 2006

[61]Boese,K. D.;Kahng,A. B. andMuddu, S. (1994) A new adaptive multi-start technique forcombinatorial global optimization.*Operations Research Letters*[online].16 (1), pp.101–113.

[62] Gendreau, M. Hertz, A., and Laporte, G. (1992), "New insertion and post-optimization procedures for the travelling salesman problem", Operations Research, forthcoming.

[63] E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan and D.B. Shmoys (eds.), The Traveling Salesman Problem. A Guided Tour of Combinatorial Optimization, Wiley, Chichester, 361-401

[64] Reinelt, G. (1989), "Fast heuristics for large geometric travelling salesman problems", Report No. 185, Institut fiir Mathematik, Universit~it Augsburg

[65] Eiselt, H.A., and Laporte, G. (1991), "A combinatorial optimization problem arising in dartboard design", Journal of the Operational Research Society 42, 113-118

[66] Bland, R.G., and Shallcross, D.F. (1989), "Large travelling salesman problems arising experiments in X-ray crystallography: A preliminary report on computation", Operations Research Letters 8, 125-128..