International Journal of Application or Innovation in Engineering & Management (IJAIEM)
Web Site: www.ijaiem.org Email: editor@ijaiem.org
Volume 3, Issue 11, November 2014 **ISSN 2319 - 4847**

Performance Evaluation of Ant Colony Optimization Algorithm and Genetic Algorithm in Travelling Salesman Problem

O. D. Fenwa[a*], I A. Adeyanju[b*] and O.O. Adeosun[c*]

[a b c]Department of Computer Science and Engineering, LAUTECH, P.M.B 4000, Ogbomoso, Nigeria.

ABSTRACT

Traveling Salesman Problem (TSP) is a well-known, popular and extensively studied problem in the field of combinatorial optimization and attracts computer scientists, mathematicians and others. Its statement is deceptively simple, but yet it remains one of the most challenging problems in operational research. It also an optimization problem of finding a shortest closed tour that visits all the given cities within the shortest time. Several optimization techniques have been used to solve the Travelling Salesman Problems such as; Ant Colony Optimization Algorithm (ACO), Genetic Algorithm (GA) and Simulated Annealing, but comparative analysis of ACO and GA in TSP has not been carried out. In this paper, an evaluation of performance was made between the Ant Colony Optimization (ACO) and Genetic Algorithm (GA) in optimizing the nearest city and distance covered by the traveler. The simulation was done and carried out on Matlab 7.10a. The results generated show that GA is a well -accepted simulator in solving the Travelling Salesman Problem, as it out performs the ACO in terms of simulation time and distance covered. Hence GA is a useful tool in solving the travelling salesman problem, as it optimizes better than the ACO.

Keywords:- Genetic Algorithm, Ant Colony Optimization, Swarm Intelligence, Pheromone

1. INTRODUCTION

The Travelling Salesman Problem (TSP) is a non-deterministic polynomial hard problem in combinatorial optimization studied in operations research and theoretical computer science. The problem was described as: there are cities and given distances between them, a travelling salesman has to visit all of them, but he does not want to spend much time on travelling, therefore we need to find the sequence of cities to minimize the travelled distance. The problem was first formulated as a mathematical problem in 1930 and is one of the most intensively studied problems in optimization. It is used as a benchmark for many optimization methods. Even though the problem is computationally difficult, a large number of heuristics and exact methods are known, so that some instances with tens of thousands of cities can be solved Several optimization techniques can be used to optimize the traveler's journey based on distance covered; some of these include Ant Colony Optimization, Genetic Algorithm, Simulated Annealing, e t c. The Ant Colony Optimization (ACO) is a family member of the Swarm Intelligence based approaches applied for optimization problems. The ACO algorithm is a probabilistic technique for solving computational problems which can be reduced to finding good paths through graphs. A multi-path data transfer is also accomplished to provide reliable network operations, while considering the energy levels of the nodes. ACO is designed to simulate the ability of ant colonies to determine shortest paths to food. Although individual ants possess few capabilities, their operation as a colony is capable of complex behavior. Real ants can indirectly communicate by pheromone information without using visual cues and are capable of finding the shortest path between food sources and their nests. The ant deposits pheromone on the trail while walking, and the other ants follow the pheromone trails with some probability which are proportioned to the density of the pheromone. The more ants walk on a trail, the more pheromone is deposited on it and more ants follow the trail. Through this mechanism, ants will eventually find the shortest path. Artificial ants imitate the behavior of real ants how they forage the food, but can solve much more complicated problems than real ants can. A search algorithm with such concept is called Ant Colony Optimization. Genetic Algorithm (GA) is a family of evolutionary algorithms based approaches applied for optimization problems using techniques inspired by natural evolution, such as inheritance, mutation, selection, and crossover. Genetic Algorithm convert the problem into a model by using chromosomes like data structure and evolve the chromosomes using selection, recombination and mutation operator [1]. GA begins with randomly selected population of chromosomes which represents the problem to be solved. An evaluation function is used to examine the "goodness" of each chromosome. The operation start from an initial population of randomly generated chromosomes population evolved for a number of generation and every time quality of an individual gradually increased. The evolution usually starts from a population of randomly generated individuals and happens in generations. In each generation, the fitness of every individual in the population is evaluated, the more fit individuals are stochastically selected from the current population, and each individual's genome is modified (recombined and possibly randomly mutated) to form a new population. The new population is then used in the next

iteration of the algorithm. Commonly, the algorithm terminates when either a maximum number of generations has been produced, or a satisfactory fitness level has been reached for the population [2]. The focus of this paper is an evaluation of performance of GA and ACO algorithms for solving Travelling Salesman Problem. It compares the optimization abilities of ACO and GA in terms of simulation time and distance covered.

2. ANT COLONY OPTIMIZATION

An artificial ant is an agent which moves from city to city on a TSP graph. It chooses the city to move to using a probabilistic function both of trail accumulated on edges and of a heuristic value, which was chosen here to be a function of the edges length. Artificial ants probabilistically prefer cities that are connected by edges with a lot of pheromone trail and which are close-by. Initially, artificial ants are placed on randomly selected cities. At each time step they move to new cities and modify the pheromone trail on the edges used this is termed local trail updating. When all the ants have completed a tour the ant that made the shortest tour modifies the edges belonging to its tour –termed global trail updating– by adding an amount of pheromone trail that is inversely proportional to the tour length. There are three ideas from natural ant behavior that we have transferred to our artificial ant colony:

(i)The preference for paths with a high pheromone level
(ii) The higher rate of growth of the amount of pheromone on shorter paths and
(iii) The trail mediated communication among ants.

Artificial ants were also given a few capabilities which do not have a natural counterpart, but which have been observed to be well suited to the TSP application: artificial ants can determine how far away cities are, and they are endowed with a working memory Mk used to memorize cities already visited (the working memory is emptied at the beginning of each new tour, and is updated after each time step by adding the new visited city [3]. At each stage, the ant chooses to move from one city to another according to some rules:

(i)It must visit each city exactly once.
(ii) A distant city has less chance of being chosen (the visibility).
(iii) The more intense the pheromone trail laid out on an edge between two cities, the greater the probability that that edge will be chosen.
(iv) Having completed its journey, the ant deposits more pheromones on all edges it traversed, if the journey is short.
(v) After each iteration trails of pheromones evaporate.

Ant Colony Optimization according to [4], is one of the most popular meta-heuristics used for combinatorial optimization (CO) in which an optimal solution is sought over a discrete search space. The well-known CO's example is the travelling salesman problem (TSP) where the search-space of candidate solutions grows more than exponentially as the size of the problem increase, which makes an exhaustive search for optimal solution infeasible [5]. The underlying idea was to use several constructive computational agents (simulating real ants). Ant's behavior is governed by the goal of colony survival rather than being focused on the survival of individuals. The behavior that provided the inspiration for ACO is the ants' foraging behavior (see Figure 2.1), and in particular, how ants can find shortest paths between food sources and their nest. When searching for food, ants initially explore the area surrounding their nest in a random manner. While moving, ants leave a chemical pheromone trail on the ground. Ants can smell pheromone [6].

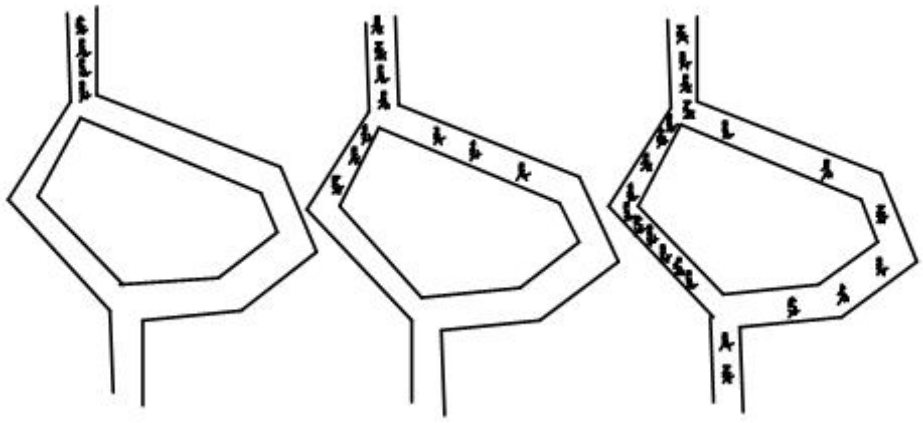

Figure 2.1 Sketch map of Ant Theory

When choosing their way, they tend to choose, in probability, paths marked by strong pheromone concentrations. As soon as an ant finds a food source, it evaluates the quantity and the quality of the food and carries some of it back to the nest. During the return trip, the quantity of pheromone that an ant leaves on the ground may depend on the quantity and quality of the food. The pheromone trails will guide other ants to the food source. It has been shown by [7] that the indirect communication between the ants via pheromone trails enables them to find shortest paths between their nest and food sources.

At each stage, the ant chooses to move from one city to another according to some rules:
i. It must visit each city exactly once.
ii. A distant city has less chance of being chosen (the visibility).
iii. The more intense the pheromone trail laid out on an edge between two cities, the greater the probability that that edge will be chosen.
iv. Having completed its journey, the ant deposits more pheromones on all edges it traversed, if the journey is short.
v. After each iteration, trails of pheromones evaporate.
As was the case in Ant System, global updating is intended to increase the attractiveness of promising route but ACS mechanism is more effective since it avoids long convergence time by directly concentrate the search in a neighbourhoods of the best tour found up to the current iteration of the algorithm. ANTS algorithm within the ACO frame-work has two mechanisms:
i. Attractiveness: The attractiveness of a move can be effectively estimated by means of lower bounds (upper bounds in the case of maximization problems) on the cost of the completion of a partial solution. In fact, if a state ι corresponds to a partial problem solution it is possible to compute a lower bound on the cost of a complete solution containing.
ii. Trail update: A good trail updating mechanism avoids stagnation, the undesirable situation in which all ants repeatedly construct the same solutions making any further exploration in the search process impossible. Stagnation derives from an excessive trail level on the moves of one solution, and can be observed in advanced phases of the search process, if parameters are not well tuned to the problem.
Pseudocode for an ACO Procedure
Begin;
Initialize the pheromone trails and parameters;
Generate population of m solutions (ants);
For each individual ant k to m:
Calculate fitness (k);
For each ant determine its best position;
Determine the best global ant;
Update the pheromone trail;
Check if termination Z true;
End;
The Model architecture is as shown in figure 2.2 below.

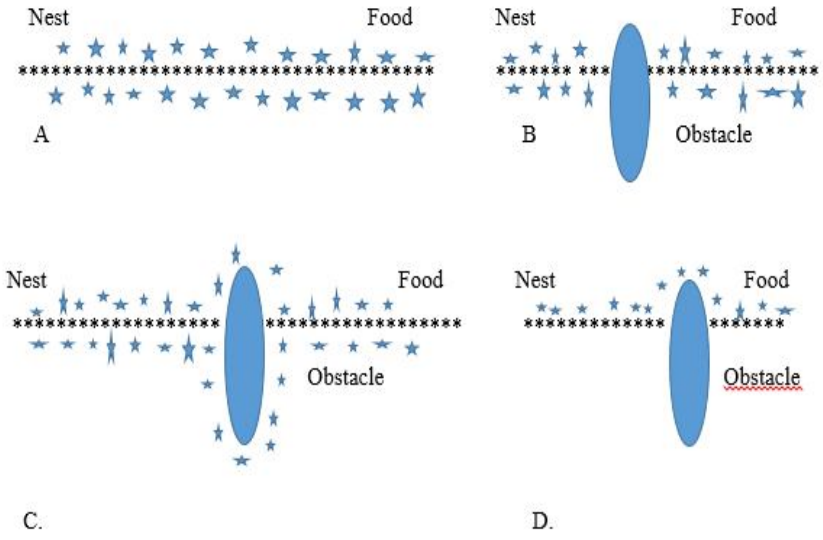

Figure 2.2: The Ant Colony Model Architecture [3]
In A: Real ants follow a path between nest and food source.
In B: An obstacle appears on the path and Ants choose whether to turn left or right with equal probability.
In C: Pheromone is deposited more quickly on the shorter path.
In D: All Ants have chosen the shorter path.

There are many different ways to translate the above principles into a computational system apt to solve the TSP. In the Ant Colony System (ACS), an artificial ant k in city r chooses the city s to move to among those which do not belong to its working memory M_k by applying the following probabilistic formula:

$s = \{argMax|\{\tau[r,u].[\cap(r,u)]\}$ if $q \leq q0$ otherwise otherwise where τ(r, u) is the amount of pheromone trail on edge (r, u), η(r, u) is a heuristic function, which was chosen to be the inverse of the distance between cities r and u, β is a parameter which weighs the relative importance of pheromone trail and of closeness, q is a value chosen randomly with uniform probability in [0,1], q_0 ($0 \leq q_0 \leq 1$) is a parameter, and S is a random variable selected according to the following probability distribution, which favors edges which are shorter and have a higher level of pheromone trail.

3. GENETIC ALGORITHM

According to [8], GA is commonly used in applications where search space is huge and the precise results are not very important. The advantage of a GA is that the process is completely automatic and avoids local minima. The main components of GA are: crossover, mutation, and a fitness function. A chromosome represents a solution in GA. The crossover operation is used to generate a new chromosome from a set of parents while the mutation operator adds variation. The fitness function evaluates a chromosome based on predefined criteria. A better fitness value of a chromosome increases its survival chance. A population is a collection of chromosomes. A new population is obtained using standard genetic operations such as single-point crossover, mutation, and selection operator. As a GA is relatively computation intensive, this chapter proposes executing the algorithm only at the base station. The proposed GA is used to generate balanced and energy efficient data aggregation trees for wireless sensor networks. The following sections present the design of the proposed GA.

3.1. Basic Description of GA

Genetic algorithm is started with a set of solutions (represented by chromosomes) called population. Solutions from one population are taken and used to form a new population. This is motivated by a hope, that the new population will be better than the old one. Solutions which are selected to form new solutions (offspring) are selected according to their fitness; the more suitable they are the more chances they have to reproduce. This is repeated until some condition (for example number of populations or improvement of the best solution) is satisfied.

Outline of the Basic Genetic Algorithm

1) **Start:** Generate random population of n chromosomes (suitable solutions for the problem).
2) **Fitness:** Evaluate the fitness f(x) of each chromosome x in the population.
3) **New population:** Create a new population by repeating the following steps until the new population is complete:
a. **Selection:** Select two parent chromosomes from a population according to their fitness (the better fitness, the bigger chance to be selected)
b. **Crossover:** With a crossover probability, crossover the parents to form a new offspring (children). If no crossover was performed, offspring is an exact copy of parents.
c. **Mutation:** With a mutation probability, mutate new offspring at each locus (position in chromosome)
d. **Accepting:** Place new offspring in a new population.
4) **Replace:** Use new generated population for a further run of algorithm
5) **Test:** If the end condition is satisfied, stop, and return the best solution in current population
6) **Loop:** Go to step 2

Traveller's Route

The traveller's route is a path incurred by the traveller in order to achieve his aim, the path taken by the traveller shows sequence by which he tends to arrive at his destination, and return back as soon as possible. The traveller is often faced with which route to take or embark upon as he has a number of choices, which include finding the minimum of the two paths, he has to embark upon.

3.2 Path by the Traveller

Two paths are presented before the traveller, which includes Path A, and Path B. These paths are synonymous to the travellers' decision, and intuitive making. The traveller has no choice in determining the best of the two paths, this is achieved by finding or checking the minimum distance available in terms of kilometre. A schematic of the paths are shown in the figure 3.1 below. The traveller is in a fix between which path is the best for her to attain optimum. These can be resolved based on the traveller's inductive reasoning or computer simulation.

3.3 Traveller's Inductive Reasoning

The traveler's inductive reasoning could be based on his personal discretion. Choices made by individuals are often as a result of emotional influence, or personal experience. Emotional influence arises as a result of the traveler's emotions

and feelings which could alter his decision making ability. The model architecture of the Traveler is as shown in figure 3.1

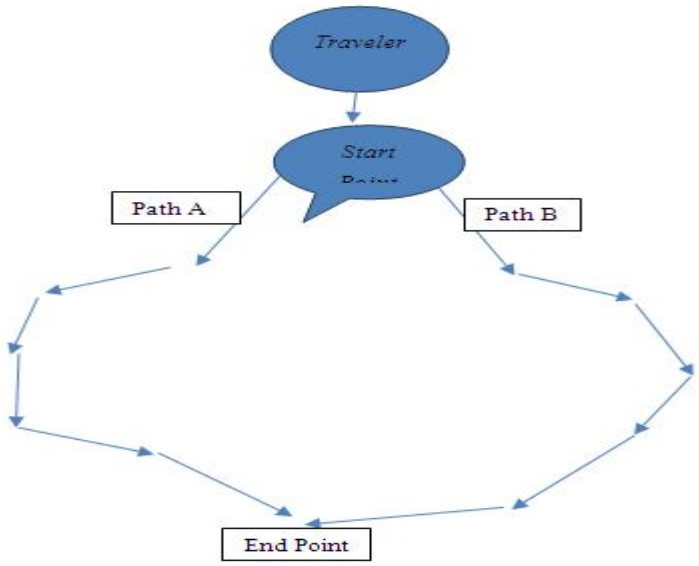

Figure 3.1 Model Architecture of the Traveler

Figure 3.1 above shows the paths to be taken by the traveler, these paths are available as the traveller starts from point one to the end point, but faced with two paths. The question of which of these paths should the traveller take to achieve minimum distance and minimum time arises. In order to achieve or answer the question the traveller has to optimize giving the constraints on time, and minimum distance. These optimization sequences can be solved using either the Ant Colony Optimization or Genetic Algorithm Optimization.

4. SYSTEM DESIGN AND IMPLEMENTATION

The system implementation was done on Matlab with a graphical user interface which is very friendly. The user can supply the number of towns, and also choose which of the simulation options he or she wants. The parameters for performance evaluation is checked, and properly analyzed. The graphical user interface of the developed system is as shown in figure 4.1 and the results generated are as shown in table 4.1 below:

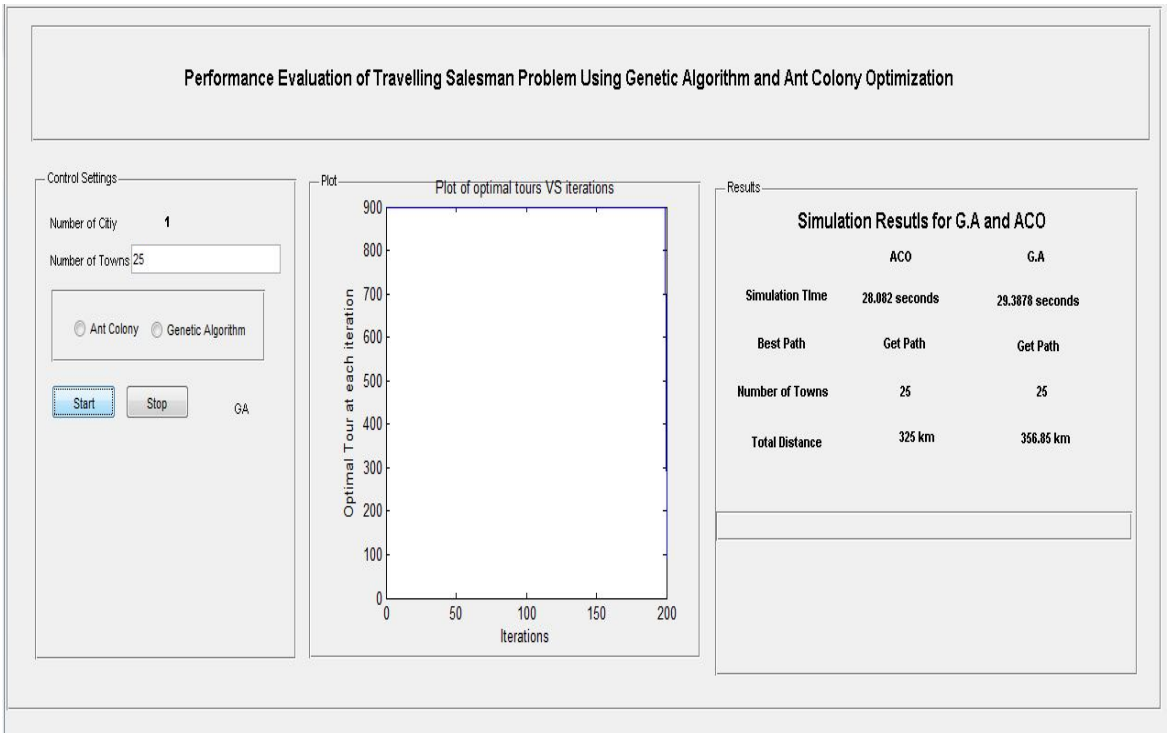

Figure 4.1 Graphical User Interface

Table 4.1 Simulation Results

S/N	Towns	Parameters	Ant Colony Optimization(ACO)	Genetic Algorithm(GA)
1	3	Total Distance	6 km	6. 07km
		Simulation Time	3.0078 seconds	16.71 seconds
2	6	Total Distance	21 km	20.57km
		Simulation Time	6.19 seconds	18.21 seconds
3	9	Total Distance	45 km	29.62km
		Simulation Time	5.42 seconds	18.10 seconds
4	15	Total Distance	120 km	92.54km
		Simulation Time	13.08 seconds	22.45 seconds
5	20	Total Distance	210 km	351.83 km
		Simulation Time	23.63 seconds	24.10 seconds
6	25	Total Distance	325 km	397km
		Simulation Time	34.43 seconds	27.56 seconds
7	30	Total Distance	465 km	508.79km
		Simulation Time	41.45 seconds	28.41 seconds
8	32	Total Distance	528 km	435.67km
		Simulation Time	44.18 seconds	29.57 seconds
9	35	Total Distance	630km	449.29km
		Simulation Time	55.19 seconds	36.12 seconds
10	40	Total Distance	820 km	508.17 km
		Simulation Time	73.41 seconds	35.31 seconds

4.2 Discussion of Results

Table 4.1 shows the simulation results of the developed system. As the results tend to have a similar trend and occurrence for each simulation sequence, the Ant Colony Optimization performs well in terms of smaller towns, as compared to the Genetic Algorithm due to the fact that it covers larger distance in smaller time, while the Genetic Algorithm favors more towns: i.e. it works in retrospect for more towns which is a weakness of the Ant Colony Optimization simulation.

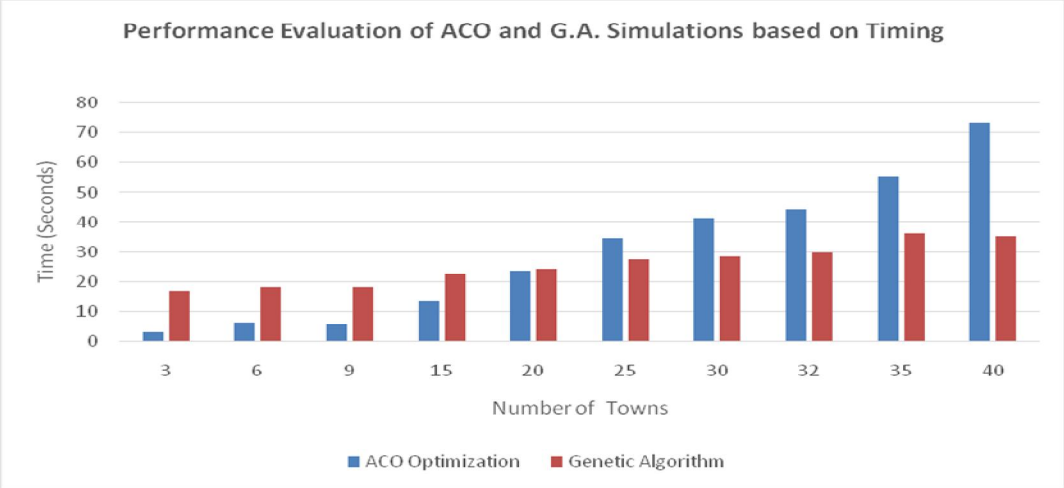

Figure 4.2 Performance Evaluation of ACO simulation and G. A simulation based on timing

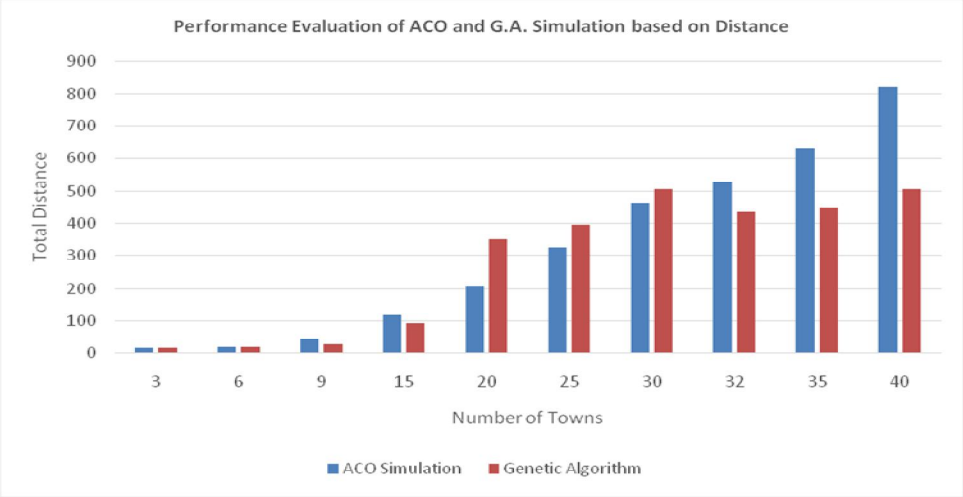

Figure 4.3: Performance Evaluation of ACO and GA Simulations based on Distance

Figure 4.2 above explicitly shows the simulation time, lower time are generated at instance and are covertly followed to the end (i.e. the sequence of timing increases rapidly with Ant Colony Optimization) which aids or support more memory consumption, the timing arises as a result of the inadequacy of the Ant to quickly locate the nearest path or the shorter path to the desired destination. For Genetic Algorithm, it is possible for the chromosome pair selection to quickly identify the nearest chromosome and fit it back to the simulation sequence. In figure 4.3 above, the total distance obtained for both ACO and GA were compared and then evaluated. The distance is a sum of all the towns the traveler is able to visit within the time frame. It is discovered in Genetic Algorithm Optimization that, the distance covered were reduced to a minimum. The minimum distance is obtained as a result of the chromosome pairing function in each population which optimizes to reduce redundancy during iteration. As a result, GA fetches out the distance and it becomes consistent while ACO keeps increasing in distance.

5. CONCLUSION AND RECOMMENDATION

In this paper, a performance evaluation was made between the Ant Colony Optimization and Genetic Algorithm in optimizing the nearest city and distance covered by the traveler. The simulation was done and carried out on Matlab 7.10a. The results generated show that GA is a well -accepted simulator in solving the travelling salesman problem as it outperforms the ACO in terms of simulation time and distance covered. Hence GA is a useful tool in solving the travelling salesman problem, as it optimizes better than the ACO. Future work could be tailored towards simulation on system with higher memory. However, other optimization techniques such as Simulated Annealing, Particle Swarm Optimization could be used for evaluation and conventional programming languages such as C#, java, and Python could also be used as a platform to check results.

REFERENCES

[1] Bhumit, Kanika and Anand(2013): Implementation of Genetic Algorithm on Intrusion Detection System to Enhance Result, International Journal for Advanced Research in Engineering and Technology, 1(5): 78-82.
[2] Angeli, Savita and Gurpa (2013): Comparing ACO and GA for Energy Efficient Coverage in WSNs, International Journal for Advanced Research in Engineering and Technology, 1(2): 64-71.
[3] Dorigo, M. and Gambardella, L.M.(1996): A study of some properties of Ant-Q, in: Proceedings of PPSN IV–Fourth International Conference on Parallel Problem Solving from Nature, H.–M. Voigt, W. Ebeling, I. Rechenberg and H.–S. Schwefel (eds.) (Springer-Verlag, Berlin) pp. 656–665.
[4] Madan, R., Cui,S., Lall,S and Goldsmith,A..J (2007). Modeling and Optimization of Transmission Schemesin Energy-Constrained Wireless Sensor Networks. IEEE/ACM Transactions on Networking, 15(6), 1359-1372.
[5] Karl, H., & Willig, A. (2005). Protocols and architectures for wireless sensor networks. Hoboken, NJ: Wiley.
[6] Wang,X., Ma,J., & Wang, S. (2009). Parallel energy-efficient coverage optimization with maximum entropy clustering in wireless sensor networks. Journal of Parallel and Distributed Computing, 69, 838-847.
[7] Ferentinos,K.P., & Tsiligiridis,T.A. (2007). Adaptive design optimization of wireless sensor networks using genetic algorithms. Computer Networks, 51, 1031-1051.
[8] Golden, B. and Stewart, W., 1985, Empiric analysis of heuristics, in The Traveling Salesman Problem, E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy-Kan and D. B. Shmoys (eds.) (Wiley, New York) pp. 207–250.