J. Appl. Environ. Biol. Sci., 6(4S)68-76, 2016

ISSN: 2090-4274
Journal of Applied Environmental and Biological Sciences
www.textroad.com

Research on Travel Salesman Problem Optimization using PSO and ACO

Hashim Ali[1], Fazl Hadi[2], Ahmadullah[3], Muhammad Haris[4], Salman[5], Naveed Ali[6]

[1, 2, 3, 4, 5]Department of Computer Science,
Abdul Wali Khan University, Mardan, KPK, Pakistan
[6] Department of Computer Science,
City University, Peshawar

Received: January7, 2016
Accepted: March 22, 2016

ABSTRACT

The traveling salesman problem (TSP) is one of the most widely studied NP-hard combinatorial optimization problems. In this paper we have proposed nature inspired ant colony optimization and Particle swarm optimization to solve travelling salesman problem. A comparative analysis is done among ant colony and particle swarm based approaches. Particle Swarm Optimization (PSO) algorithm was developed under the inspiration of behavior and laws of bird flocks, fish schools and human communities. Compared with the Ant colony optimization algorithm inspired from ant colonies, PSO algorithm has high convergence speed. In this paper, the disadvantages of Ant colony algorithm like being trapped easily into a local optimum are discussed in detail, we use the PSO algorithm to solve the TSP and the experiment results show that PSO algorithm is effective for the this problem. All the implementation are done keeping some factors like time complexity and cost efficiency for performance evaluation.

KEYWORD: Travel salesman problem (TSP), Particle swarm optimization (PSO), Ant colony optimization (ACO), cost, Time complexity.

1. INTRODUCTION

The traveling salesman problem (TSP) [1] is one of the most widely studied NP-hard combinatorial optimization problems. Its statement is deceivingly simple, and yet it remains one of the most interesting problems in Operational Research. TSP can be described as: Give a shortest path that covers all cities along. Let G = (V; E) be a graph where V is a set of vertices and E is a set of edges. Let C= (c_{ij}) be a distance (or cost) matrix associated with E. The TSP requires determination of a minimum distance circuit (Hamiltonian circuit or cycle) passing through each vertex once and only once. C is said to satisfy the triangle inequality if and only if cij $+c_j^k + \geq c_i^k$ for i, j, k ∈V.

The attempt in the research of computer technology is to develop algorithms inspired by insect behavior to solve optimization problems. Due to its simple description and wide application in real practice such as Path Problem, Routing Problem and Distribution Problem, it has attracted researchers of various domains to work for its better solutions. Those traditional algorithms such as Cupidity Algorithm, Dynamic Programming Algorithm, are all facing the same obstacle, which is when the problem scale N reaches to a certain degree, the so-called "Combination Explosion" will occur. For example, if N = 50, then it will take 5×10^{48} years under a super mainframe executing 100 million instructions per second to reach its approximate best solution. A lot of algorithms have been proposed to solve TSP [2] [3] . Many of them (based on dynamic programming or branch and bound methods) gives the global optimal solution. Other algorithms are heuristic ones, which are faster, but they do not surety to gives the optimal solutions. There are famous algorithms based on 2-opt or 3-opt change operators, Lin-Kerninghan algorithm (variable change) as well algorithms based on greedy principles (nearest neighbor, spanning tree, etc). The TSP was also approached by various modern heuristic methods, like simulated annealing, evolutionary algorithms and tabu search, even neural networks.

Particle Swarm Optimization (PSO) algorithm was an intelligent technology first presented in 1995 by Eberhart and Kennedy, and it was developed under the inspiration of behavior laws of bird flocks, fish schools and human communities [4]. If we compare PSO with ACO Algorithms, ACO have a drawback of complexity of implementation and computation time penalty. It has also been shown that, although adding a local searcher is a good approach in the majority of cases, it may prevent ACO from finding the optimal solution [5], According to this reference we can say that ACO is better for small number of cities. PSO achieves its optimum solution by starting from a group of random solution and then searching repeatedly. Once PSO was presented, it invited widespread concerns among scholars in the optimization fields and shortly afterwards it had become a studying focus within only several years. A number of scientific achievements had emerged in these fields [6] [7]. PSO was proved to be a sort

* **Corresponding Author:** Hashim Ali, Department of Computer Science, Abdul Wali Khan University, Mardan, KPK, Pakistan. hashimali@awkum.edu.pk

of high efficient optimization algorithm by numerous research and experiments [8] . PSO is a meta-heuristic as it makes few or no assumptions about the problem being optimized and can search very large spaces of candidate solutions. However, metaheuristics such as PSO do not guarantee an optimal solution is ever found. More specifically, PSO does not use the gradient of the problem being optimized, which means PSO does not require that the optimization problem be differentiable as is required by classic optimization methods such as gradient descent and quasi-Newton methods. PSO can therefore also be used on optimization problems that are partially irregular, noisy, change over time, etc. This paper compare PSO with ACO being easily trapped into a local optimum and proposed that PSO algorithm which proves to be more simply conducted and with more efficient global searching capability, then use the PSO algorithm for engineering optimization field.

Ant colony optimization (ACO) is one of the most successful techniques in the wider field of swarm intelligence. Many research works have been devoted to ant colony optimization techniques in different areas. It is a relatively novel meta-heuristic technique and has been successfully used in many applications especially problems that belong to the combinatorial optimization. ACO inspired by the foraging behavior of real ant was first introduced by dorigo and his colleagues [9] [10] in early 1990s and has become one of the most efficient algorithms for TSP. ACO is based on the pheromone trail laying and following behavior of some ant species, a behavior that was shown to allow real ant colonies to find shortest paths between their colony and food sources. These ants deposit pheromone on the ground in order to mark some favorable path that should be followed by other members of the colony. The ants move according to the amount of pheromones, the richer the pheromone trail on a path is, the more likely it would be followed by other ants. So a shorter path has a higher amount of pheromone in probability, ants will tend to choose a shorter path. Artificial ants imitate the behavior of real ants how they forage the food, but can solve much more complicated problem than real ants can. Ant colony optimization exploits a similar mechanism for solving optimization problems.

2. RELATED WORK

In current years, since the TSP is a well-known ground for challenging optimization techniques, researchers are interested in various fields such as artificial intelligence, biology, mathematics, physics, and procedures research allocate themselves to trying to find the effective methods for solving the TSP, such as genetic algorithms (GAs) [11], ant colony optimization (ACO) [12], simulated annealing (SA) [13], evolutionary algorithms (EA) [14]neural networks (NN) [15], particle swarm optimization (PSO) [16], memetic computing [17], etc. Besides, there are many practical applications of the TSP in the real world [18] [19], such as data association, vehicle routing (with the additional constraints of vehicle's route, such as capacity's vehicles), data transmission in computer networks, job scheduling, DNA sequencing, drilling of printed circuits boards, clustering of data arrays, image processing and pattern recognition, analysis of the structure of crystals, transportation and logistics.

A hybrid approach that joins PSO, Genetic Algorithms and Fast Local Search is presented by Machado & Lopes [20] for the TSP. The positions of the particles represent TSP tours as permutations of |N| cities. The value assigned to each particle (fitness) is the rate between a constant Dmin and the cost of the tour represented in the particle's position. The hybrid PSO is applied to the following symmetric TSP benchmark instances: pr76, rat195, pr299, pr439, d657, pr1002, d1291, rl1304, d2103.

Goldbarg [21] present a PSO algorithm for the TSP where the idea of distinct velocity operators is introduced. The velocity operators are defined according to the possible movements a particle is allowed to do. This algorithmic proposal obtained very promising results. It was applied to 35 benchmark TSP instances with 1 to 80 cities. The results were comparable to the results of tate-of-the-art algorithms for the TSP.

ACO has been the attraction point of researchers over last decade. The performance of ACO depends on the termination condition being selected. In [22] they have predicted best termination condition for ACO. This prediction is done from the solved instances of the problem having best results. Then this predicted termination condition is applied over new instances of problem. In [23] ACO has been applied on cloud task scheduling which is also a NP-Hard problem. ACO is used to predict to which VM incoming cloudlet should be submitted to balance the load and efficient utilization of cloud resources. In [24] authors have applied ACO on dynamic traffic planning approach to balance the traffic among the possible travel paths available between a source and destination.

3. Problem statement:

The ant colony algorithm using pheromones and heuristic information to find the optimal path. ACO is better for small problem space and for minimum number of cities. The problem in ACO is that ACO trap in local optimum and do not find the optimal path. The genetic algorithm is also trapped in local optimum.

4. Proposed Solution:

We are using the new algorithm to solve the TSP problem. This new algorithm is efficient in optimizing TSP problem. This new algorithm gives more efficient result than ACO. The new algorithm is best on the basis of cost and time complexity.

4.1 Particle Swarm Optimization:

A basic variant of the PSO algorithm works by having a population swarm) of candidate solutions (particles). These particles move around in the search space. The movements is guided by their own best known position in the search-space as well as the entire swarm's best known position. As the particles finds better positions, they will then participate to monitor the movements of the swarm. The process is repeated and by doing so it is predicted, but not guaranteed, that a satisfactory solution will eventually be discovered. Formally, Let f: Rn → R be the cost function which must be minimized. The function takes a candidate solution as argument in the form of a vector of real numbers and produces a real number as output which indicates the objective function value of the given candidate solution.

PSO was proposed under the stimulus of bird flock arrival during the sequence of finding food and then be used in the optimization problems. In PSO, each optimization problem solution is taken as a bird in the searching space and it is called "particle". Every particle has a fitness value which is determined by target functions and its velocity which determines the destination and distance. All particles search in the solution space for their best positions and the positions of the best particles in the swarm. Initially PSO consist a group of random particles (random solutions), and then the optimum solutions are found by performing repeated searching. In every iteration, a particle will follow two bests to renew itself: the best position found for a particle called pbest; the best position found for the whole swarm called gbest. All particles will determine following steps through the best experiences of individuals themselves and their companions.

For particle id, its velocity and position renewal, the mathematical algorithm is as follows:

In the classical PSO algorithm, each particle

- has a position and a velocity
- knows its own position and the value associated with it
- knows the best position it has ever achieved, and the value associated with it
- knows its neighbors, their best positions and their values

The movement of a particle is a composed of three possible choices

- To follow its own way
- To go back to its best previous position
- To go towards its best neighbor's previous or present position.

Mathematically,

$$v_{id} = w * v_{id} + \alpha_1 * rand() * (P_{best} - X_{i-1}) + \alpha_2 * rand() * (G_{best} - X_{i-1}) \quad \ldots\ldots\ldots\ldots (1.0)$$

$$X_i = X_{i-1} + v_{id} \quad \ldots\ldots\ldots\ldots\ldots\ldots (2.0)$$

α_1 is a cognitive coefficient that quantifies how much the particle trusts its experience, α_2 is a social coefficient that quantifies how much the particle trusts its best neighbor. Both α 1 and α 2 are also called accelerating factors. Rand1 and rand2 are random numbers.

In equation (1) ω represent inertia weight, it is a proportion factor that is concerned with former velocity, $0< \omega<1$, α_1 and α_2 are constants and are called accelerating factors, normally $\alpha_1 = \alpha_2 = 2$, rand () are random numbers, X_{id} represents the position of particle id; Vid represents the velocity of particle id; P_{id}, P_{gd} represent separately the best position particle id has found and the position of the best particles in the whole swarm.

In equation (1), the first part represents the former velocity of the particle. It enables the particle to possess expanding tendency in the searching space and thus makes the algorithm more capable in global searching; the second part is called cognition part. It represents the process of absorbing individual experience knowledge on the part of the particle; the third part is called social part, it represents the process of learning from the experiences of other particles on the part of certain particle, and it also shows the information sharing and social cooperation among particles. The flow of PSO can briefly describe as following:

First, to initialize a group of particles, e.g. to give randomly each particle an initial position Xi and an initial velocity Vi, and then to calculate its fitness value f. In every iteration, evaluated a particle's fitness value by analyzing the velocity and positions of renewed particles in equation (1) and (2). When a particle finds a better position than previously, it will mark this coordinate into vector P1, the vector difference between P1 and the present position of the particle will randomly be added to next velocity vector, so that the following renewed particles will search around

this point, it's also called in equation (1) cognition component. The weight difference of the present position of the particle swarm and the best position of the swarm Pgd will also be added to velocity vector for adjusting the next population velocity. This is also called in equation (1) social component. These two adjustments will enable particles to search around two bests. The most obvious advantage of PSO is that the convergence speed of the swarm is very high, scholars like has presented proof on its convergence. In order to verify the convergence speed of the PSO algorithm.

Algorithm 1: Pseudo code for PSO

1. Initialization

Parameters and size of the swarm (S)
Randomly initialize particles positions and velocities
For each particle, let pbest = x
Calculate f(x) of each particle
Calculate gbest

2. While (termination criterion is not met)

For i = 1 to S
Calculate the new velocity using equation (1)
Calculate the new position using equation (2)
Calculate f(x) of each particle
If (f(x) < f(pbest)) pbest = x
If (f(pbest) < f(gbest)) gbest = pbest

3. Show the best solution found gbest

4.2 Ant colony optimization:

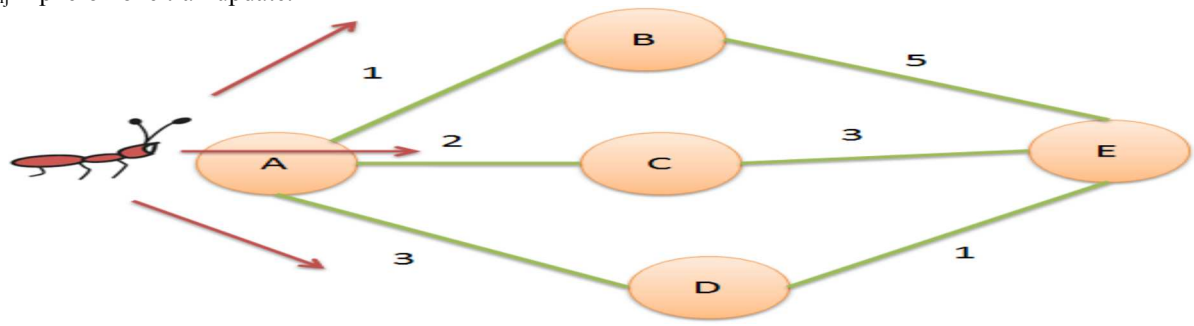
The ACO algorithm was developed by Dorigo in 1992 in his PhD thesis as an optimization method inspired by ant colony behaviors, where the author examined behaviors of ants. The Ant Colony Optimization techniques has emerged recently as a relatively novel meta-heuristic for hard combinational optimization problems. It 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 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. Figure1 shows how the ants find the shortest path.

In the TSP the traveling salesman tries to find complete tour with minimum length that visits every city exactly once. The selection of cities to which an ant moves is based on the distance and the amount of pheromones between the cities. This algorithm is iteratively repeated and the shortest route it finds is taken as the best solution. The selection of a city, j, to which an ant in another city, i, will move in iteration t, is based on a probability, P_{ij}, defined in the Eq.

$$p_{ij}^{k}=[\tau_{ij}]^{\alpha}[\eta_{ij}]^{\beta}/\Sigma l\in N_i^k[\tau i,l]^{\alpha}[\eta_{j},l]^{\beta} \quad , \quad \Omega i \quad i\ f\ j\in N_i^k$$

p_{ij}^{k} = Probability with which ant k moves from i node to j.
η_{ij}^{k} = heuristic information.
τ_{ij}^{k}= pheromone trail update.



Where represents the amount of pheromone between the city (i) to the city (j); Ωi is the set of cities allowed for the kth ant in step t. The constants α and β are parameters which control the importance of the pheromone versus the heuristic information η_{ij}, defined by the Eq. (i):

$$\eta_{ij=1/d_{ij}} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \text{(i)}$$

Where d_{ij} is the distance between cities i and j. The update of the amount of pheromones on the inter-city roads in the (t + 1)th iteration is defined by the Eq. (ii). The equation considers the effect of evaporation and the contribution of the kth ant. It is parameterized by the evaporation rate, ρ and m is the number of ants.

$$\tau ij(t+1)=\rho \cdot \tau ij(t)+\Delta \tau ij \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (ii)$$

Where defines the amount of pheromones, laid on edge (i, j) by the kth ant; it is given by the Eq. (iii):

$$\Delta\tau_{ij}^{k} = 1/L_K \qquad \text{if } (i,j) \in \text{bestTour.} \ldots\ldots (iii)$$

Where the tour length of the kth ant, bestTour is a set of shortest path found within t iterations. Equation (iv) gives the information of pheromone with the total length of the tour.

$$\Delta \tau^{K}_{ij} = \begin{cases} Q / L_K & \text{if any } K \text{ travel on edge}(i,j) \\ 0 & \text{Otherwise} \end{cases} \qquad \ldots\ldots\ldots\ldots\ldots\ldots (iv)$$

5. Graphs and discussion:

5.1 Performance Parameters

There are different kinds of parameters for the performance evaluation of the optimization techniques. These have different behaviors of the overall performance in scenario. We will evaluate two parameters for the comparison of our study on the overall performance. These parameters are time and cost for optimization techniques evaluation. These parameters are important in the consideration of evaluation of the optimization techniques in finding optimal path in different cities. These techniques need to be checked against certain parameters for their performance.

5.2 Time complexity

This is the time that an algorithm takes to complete the generation of result. This time is expressed in sec. Hence all the delays in the execution of algorithms are called time complexity.

5.3 Tour cost

In TSP we have weighted and fully connecting graphs, so the weight of edge represent the distance between to cities. The tour is the set of fully connected cities from origin O to destination D, The total distance form origin O to destination D is known as tour cost. It can be expressed in km.

5.4 Simulated Scenarios

We simulate five scenarios. In first scenario the number of cities varies in the range of 1 to 10, in the second scenario 1 to 30 cities, in third scenario the no. of cities varies from 1 to 80. All the simulations show the required results. Under each simulation we check the behavior of Ant colony and particle swarm optimization. We get multiple graphs from simulations like first we get for cost, and second is for the time. Main goal of our simulation was to model the behavior of the optimization techniques.

5.4.1 Scenario A

In scenario A, the PSO and ACO are analyzed for cities in the range of 1 to 10.The variation of cost and iteration is given in the table below.

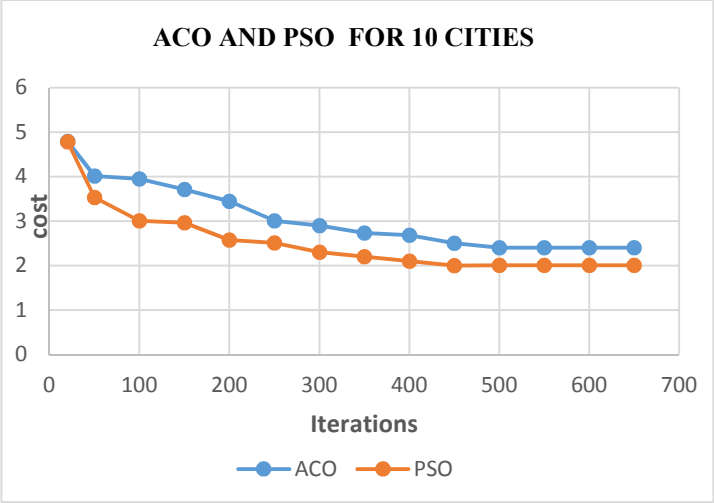


Figure 5.1 Cost Statistics (1-10) cities

Iterations	ACO	PSO
20	4.7922	4.78
50	4.00859	3.53
100	3.9504	3.0034
150	3.7104	2.96
200	3.4452	2.57
250	3.0034	2.509
300	2.9	2.301
350	2.73	2.2
400	2.678	2.1
450	2.5	2.001
500	2.4	2.0023
550	2.4	2.0034
600	2.4	2.0035
650	2.4	2.0076

table of 1-10 cities

Time complexity

In Figure comparison of cities in terms of time is shown. It is noted that the PSO algorithm has taken less time compare to ACO algorithm. The no iteration and time taken by given algorithms are given and presented on the table below.

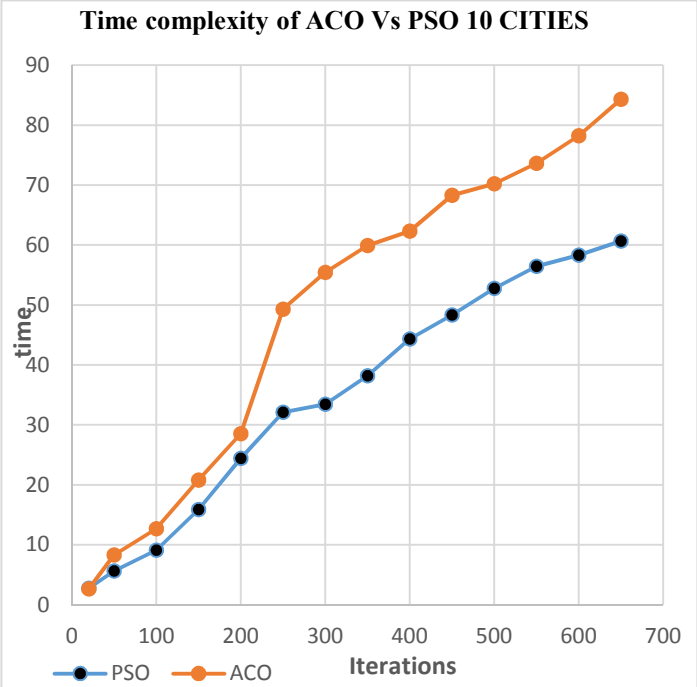


Figure 5.2 Time complexity graph of 10 cities

ITIERATIONS	PSO	ACO
20	2.731909	2.641843
50	5.648032	8.283139
100	9.09483	12.66866
150	15.85932	20.77037
200	24.40235	28.54499
250	32.1	49.28
300	33.45	55.43
350	38.21	59.89
400	44.32	62.31
450	48.32	68.31
500	52.76	70.211
550	56.45	73.65
600	58.321	78.22

Table of 10 cities

5.4.2 Scenario B

In scenario second, selected optimization techniques analyzed for cities in the range of 1 to 30. The variation of cost and iteration is given below in the table.

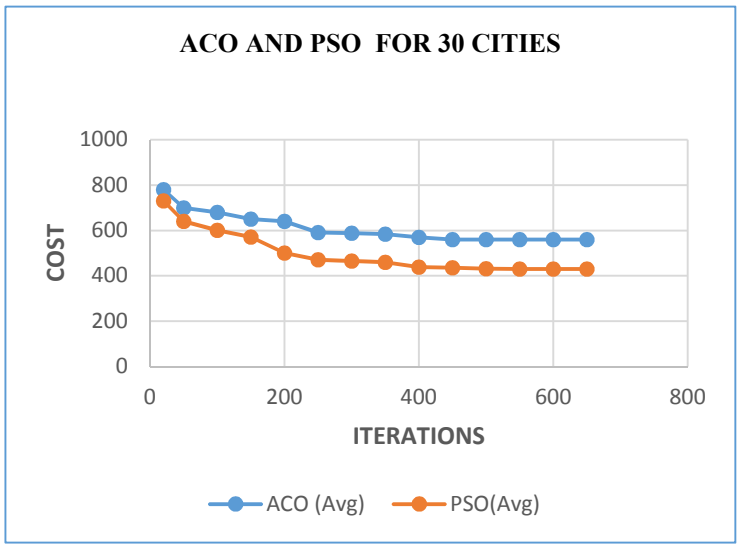


Figure 5.3 Cost Statistics (1-30) cities

Iterations	ACO (Avg)	PSO(Avg)
20	780.356	730.408
50	700.105	640.21
100	680.134	600.407
150	650.934	570.707
200	640.402	500.574
250	590.713	470.48
300	588.78	464.789
350	584.342	460.345
400	570.43	438.9
450	560.48	435.9
500	560.45	430.97
550	560.48	430.56
600	560.48	430.56
650	560.48	430.56

Table of 1-30 cities

Time complexity

In comparison of cities in terms of time is shown. It is noted that the PSO algorithm has taken less time compare to ACO algorithm. The no iteration and time taken by given algorithms are given and presented on the table below.

Table of 30 cities

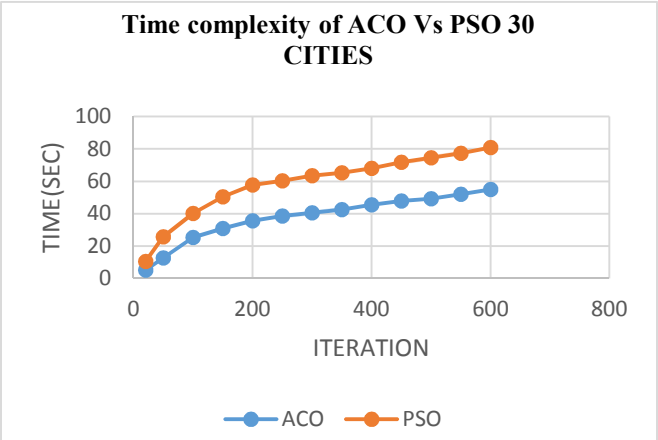


Figure 5.4 Time complexity of (1-30) cities

ITIERATIONS	ACO	PSO
20	10.43	5.06
50	25.75	12.71
100	40.07	25.22
150	50.38	30.78
200	57.82	35.51
250	60.34	38.54
300	63.56	40.45
350	65.33	42.56
400	67.99	45.41
450	71.76	47.76
500	74.43	49.33
550	77.34	51.99
600	80.76	54.87

5.4.3 Scenario c

In third scenario, selected optimization techniques analyzed for cities in the range of **1** to **80**. The variation of cost and iteration is given below in the table.

Figure 5.5 Cost Statistics (1-80) cities

Iterations	ACO	PSO
20	1030	1015
50	1019	1003.3
100	1008.9	993
150	1005	987.9
200	992.21	971.78
250	987.9	965.9
300	981.01	960.01
350	974.9	952.98
400	970.98	945.98
450	968.9	940.35
500	968.9	940
550	968.5	940
600	968.5	940
650	968.5	940

Table of 80 cities

Time complexity:

In ***Figure 5.6*** comparison of cities in terms of time is shown. It is noted that the PSO algorithm has taken less time compare to ACO algorithm. The no iteration and time taken by given algorithms are given and presented on the table below.

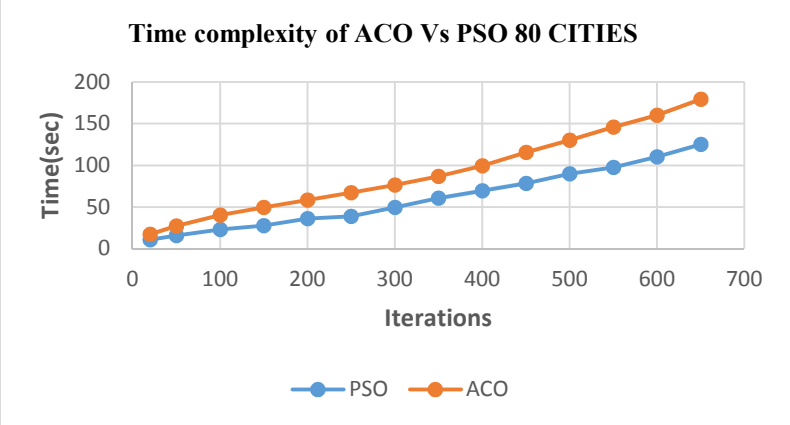


Figure 5.6 time complexity of (1-80) cities

Iteration	PSO	ACO
20	10.62	17.12
50	15.52	27.21
100	22.83	40.39
150	27.65	49.52
200	35.95	58.192
250	38.88	67.33
300	49.66	76.34
350	60.54	86.89
400	69.33	99.45
450	78.44	115.45
500	89.87	130.34
550	97.34	145.99
600	110.34	159.98
650	125.24	179.24

Table of 1-80 cities

Conclusion:

The performance of particle swarm optimization and Ant colony optimization are evaluated using three different types of simulations. In Optimization techniques cost and time are essential parameters. This paper has presented the analysis of existing optimization techniques and their critical study. We have selected certain parameters associated with cost and time. Optimization techniques such as Particle swarm optimization (PSO) and Ant colony optimization (ACO) according to those parameters. In all scenarios we have observed that PSO gives better results than ACO with respect to cost and time. We noticed that PSO generates the better solution compare to ACO while increase in number of iterations and cities. The PSO is better than ACO because the PSO using g_{best} parameter. From simulations we observed that ACO does not provides the better results on the basis of cost compare to PSO. Increase the number of cities and iterations the ACO tries to generate different solutions but not optimal solution compare to PSO. The graphs obtained from simulations shows that PSO is quick such that it executes in time less than ACO

execution time. In addition the increase in number of iterations and cities increases the execution time of both algorithms. Overall we have found PSO as more efficient with respect to time and cost compare to ACO.

REFERENCES

[1] Xuesong Yan1, Can Zhang1 , Wenjing Luo1, Wei Li1, Wei Chen1 and Hanmin Liu2, "Solve Traveling Salesman Problem Using Particle Swarm Optimization Algorithm," *IJCSI International Journal of Computer Science Issues,* pp. 264-271, November 2012.

[2] Tao Guo and Zbigniew Michalewize, "Inver-Over operator for the TSP," *In Parallel Problem Sovling from Nature(PPSNV),* pp. 803-812., 1998.

[3] Onwubolu, G.C. & Clerc, M, "Optimal path for automated drilling operations by a new heuristic approach using particle swarm optimization," *International Journal of Production Research,* vol. 42, no. 3, p. 473–491, 2004.

[4] J. Kennedy and R. C.Eberhart, "Particle Swarm Optimization," *IEEE International Conference on Neural Networks,* pp. 1942-1948, 1995.

[5] Neumann, F., Sudholt, D., and Witt, C, "Rigorous analyses for the combination of ant," *Springer-Verlag,* vol. 5217, p. 132–143, 2008.

[6] Clare M, Kennedy J, "The Particle Swarm Explosion ,Stability, and Convergence in a Multidimensional Complex Space," *IEEE Trans. on Evolution2ary Computation,* vol. 6, no. 1, pp. 58-73, 2002.

[7] J.Kennedy, "The particle swarm: social adaptation of knowledge," *In Proc. IEEE Conf. on evolutionary computation,* pp. 3003-3008, 1997.

[8] E. Oscan and C. K.Mohan, "Analysis of A Simple Particle Swarm Optimization System," *Intelligence Engineering Systems Through Artificial Neural Networks258,* pp. 253-258, 1998.

[9] C. M. Dorigo, V. Maniezzo, and A. Colorni, "The ant system:Optimization by a colony of cooperating agents," *IEEE Transactions on System, Man, and Cybernetics,* vol. 26, no. B, pp. 29-41, 1996.

[10] C.Blum, "Ant Colony Optimization: Introduction and recent trends," *ScienceDirect,* vol. 2, pp. 353-373, 2005.

[11] Hua Z, Huang F, "A variable-grouping based genetic algorithm for large-scale integer programming," *Inf Sci,* vol. 176, no. 19, p. 2869–2885, 2006.

[12] Ellabib I, Calamai P, Basir O, "Exchange strategies for multiple ant colony system," *Inf Sci ,* vol. 177, no. 5, p. 1248–1264, 2007.

[13] Lo CC, Hus CC, "Annealing framework with learning memory," *IEEE Trans Syst Man Cybern Part A ,* vol. 28, no. 5, p. 1–13, 1998.

[14] Shen G, Zhang YQ, "A new evolutionary algorithm using shadow price guided operators," *Appl Soft Comput ,* vol. 11, no. 2, p. 1983–1992, 2011.

[15] Masutti TA, de Castro LN, " A self-organizing neural network using ideas from the immune system to solve the traveling salesman problem," *Inf Sci ,* vol. 179, no. 10, p. 1454–1468, 2009.

[16] Onwubolu GC, Clerc M, "Optimal path for automated drilling operations by a new heuristic approach using particle swarm optimization," *Int J Prod Res ,* vol. 42, no. 3, p. 473–491, 2004.

[17] Acampora G, Gaeta M, Loia V, "Combining multi agent paradigm and memetic computing for personalized and adaptive learning experiences," *Comput Intell J ,* vol. 27, no. 2, p. 141–165, 2011.

[18] Marinakis Y, Marinaki M, Dounias G, "A hybrid particle swarm optimization algorithm for the vehicle routing problem," *Eng Appl Artif Intell ,* vol. 23, no. 4, p. 463–472, 2010.

[19] Banaszak D, Dale GA, Watkins AN, Jordan JD, "An optical technique for detecting fatigue cracks in aerospace structures," *In: Proc 18th ICIASF,* p. 1–7, 2009.

[20] Machado, T.R. & Lopes, H.S, "A hybrid particle swarm optimization model for the traveling salesman problem," *In:Natural Computing Algorithms,* pp. 255-258., 2005, .

[21] Goldbarg, E.F.G; Souza, G.R. & Goldbarg, M.C, "Particle swarm for the traveling salesman problem," *Proceedings of the EvoCOP ,* vol. 3906, pp. 99-110, 2006.

[22] M. Veluscek, T. Kalganova, P. Broomhead , "Improving Ant Colony Optimization Performance through Prediction of Best Termination Condition," *In IEEE ,* 2015.

[23] Medhat A. Tawfeek, Ashraf El-Sisi, Arabi E. keshk,Fawzy A. Torkey , "Cloud Task Scheduling Based on Ant Colony Optimization," *in IEEE ,* 2013.

[24] Jerry Kponyo , Yujun Kuang , Enzhan Zhang , "Dynamic Travel Path Optimization System Using Ant Colony Optimization," *in IEEE ,* 2014.