DOI: http://dx.doi.org/10.26483/ijarcs.v8i7.4244

ISSN No. 0976-5697

Volume 8, No. 7, July – August 2017

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

ACO-SA: ENHANCED OPTIMIZATION FOR TSP

Sudeepta Thukral
Department of Computer Engineering
U.I.E.T., Kurukshetra University Kurukshetra
Kurukshetra, INDIA

Chander Diwaker
Department of Computer Engineering
U.I.E.T., Kurukshetra University Kurukshetra
Kurukshetra, INDIA

***Abstract*:** Traveling salesman problem (TSP) is traditional combinatorial-optimization problem and a NP-problem in operation research. Swarm intelligence (SI) algorithms be capable of efficiently achieve best tours with minimum lengths. ACO is type of probability technology use to get optimal path in the graph. Through the analysis on the main reasons resulting in low convergence rate to overcome it simulated annealing optimization are used. In this, new hybrid ACO-SA algorithm for solving TSP depended on ACO with SA optimization technique which avoids trapping in the local-minima points. Experiments have performed with data set obtained from TSPLIB and contrast the new results of proposed method with existing methods. The results exemplify that both the average cost and number of iteration to the best known solution of proposed method are better than existing methods.

Keywords: Travelling Salesman Problem; Ant Colony Optimization; Simulated annealing

I. INTRODUCTION

TSP is the famous problem in combinatorial optimization [1].TSP is easy to explain but hard to solve. The first attempt of TSP was done in 1759 by Euler whose problem was that to traverse a knight lying on chess board just once. TSP means to find minimum path between given numbers of cities, but the condition is to visit every city only one time and arrive back to initial city [2] as shown in Fig.1. TSP is a NP hard problem [3]. Its solution search space increase exponentially with increment of the input size and its computational complex is O(*n*!) .The smart optimization algorithms which solve TSP usually include ACO, GA and SA etc. These algorithms had major advantages and run rapidly than traditional exact algorithms. But, the speed is frequently impossible to meet people's need as the scale of TSP increase gradually. In TSP, person travel N cities also appear back to initial city with the nominal cost. A completed graph G = (N, E) is used for representing TSP[2], in which N is all the n cities with E is all edges (paths) which fully linking all cities. Every edge $(i,j) \in E$ is allocated cost *dij*, which is distance among city i and city j. *dij* can defined in Euclidean space as shown below in (1):

$$dij = \sqrt{(xi - xj)^2 + (yi - yj)^2} \quad (1)$$

[2]

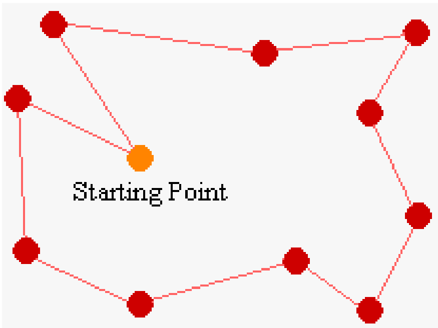


Figure 1. Graphical illustration of TSP

II. APPLICATIONS OF TSP

TSP has wide number of application. Some of them are expressed below [4]:-

(i) Transportation: It includes school bus routes, delivering meals and service calls etc [3].
(ii) Manufacturing: In industrial robots which drills the holes in print circuit boards [5].
(iii) VLSI (microchip) layout: It is utilized to design the layout.
(iv) Communication: The planning of new telecommunication networks [6].
(v) Genome Sequencing: By arranging DNA fragments in sequence.
(vi) Job Sequencing: sequencing jobs so as to minimise total set-up time between jobs.
(vii) Overhauling gas-turbine engines: It is used in aircraft [2].
(viii) X-Ray crystallography: It is used in detecting structure of crystalline material.

III. RELATED WORK

Zar Chi Su Su Hlaing and May Aye Khine [7] presented the improved ACO through two highlights. First, candidate set approach is implemented to fast convergence speed. Second, dynamic revising rule for heuristic parameters depended on the entropy and the emergence of local search solution to improved presentation in solving the TSP. *Zhao et al* [8] presented pheromone increase model called as ant constant, that maintain ants energy conversation. To represented the pheromone variation of the different candidate paths. The diffusion model of pheromone was based on info-fountain of path, to return potency field of pheromone diffusion and strengthened the collaboration among ants. *Zbigniew SWiatnicki* [9] focused on ACO algorithms. The algorithm used the grouping of two global pheromone update heuristics - BSF (best-so-far route) and IB (iteration-best route) showed high effectiveness. *Zalilah Abd Aziz* [3] designed a generalized heuristics method (hyper-heuristics) depended on ACO algorithms for resolving the TSP. It involved two pheromones updating actions; local plus global update. The global inform will employ the finest solution originate at the present iteration for updating the pheromones trails and local updates are performed subsequent to every ant performs a tour. *Yang and Wang* [6] constructed an enhanced ACO algorithm to solve TSP. By changing sum of information and searching for the optimal parameters, it can rate up convergence velocity. *Hassan Ismkhan* [10] proposed effective heuristics used for ACO (ESACO) whose performance was quick, accurate and applicable to solve very large instances up to 20000. *Abdulqader M. Mohsen* [11] proposed a new hybridized algorithm, annealing elitist ant-system having mutation operators for TSP. It integrated advantages of the ACO, the SA, and the mutation operator to solve TSP. The foundation of algorithm was depending on ACO. SA and mutation operator were utilized to increase ants population diversity from time to time and the local search was used to exploit the current search area efficiently.

IV. ALGORITHM DESIGN

The hybrid ACO-SA algorithm has designed for determining the finest solution for TSP. It contains some features of ACO and some of SA. The objective of this algorithm is to improve the performance of ACO by preventing it to get trapped in local-minima. This algorithm increases the convergence- rate of ACO by using SA. The general idea about the ACO, the SA and the projected algorithm is presented in the following subsections.

A. *Ant Colony Optimization*

ACO is population-based meta-heuristic algorithm which was inspired by the foraging activities of true ants when searching for shortest path as of the food source to nests[12]. Analogically, the artificial ants look for good solutions iteratively in several generations. In all generation, every ant constructs its possible solution gradually guided by the transition rule which is function of the artificial pheromone and distance between two cities [13]. After that, ant leaves an amount of pheromone trails on edges of its fulfilled tour. In next generation, the ants are fascinated by that pheromone trails. Therefore, this will guide the search in search space towards superior quality solutions [11]. TSP is similar to foraging activities of true ants in nature. Therefore, applying ACO to solve TSP will be very simple. The pheromone value τ_{ij} with edge ij connecting city i with city j has updated as given away in (2):

$$\tau_{ij} = (1-\rho)\,\tau_{ij} + \sum_{k=1}^{m} \Delta\square_{ij}^{k} \qquad (2)$$

[3]

The parameter ρ is parameter of pheromone-evaporate rate, the total ants are m, $\Delta\square_{ij}^{k}$ be the amount of pheromone left on edge (i,j) by ant k shown in (3):

$$\Delta\square_{ij}^{k} = \begin{cases} Q/Lk & \text{edge in ant k`s tour} \\ 0 & \text{otherwise} \end{cases} \qquad .(3)$$

[14]

where Q be the constant, L_k is length of tour which is constructed by ant k during current loop[14]. The selection is determined by the probability, *Pij*, given in (4).

$$p_{ij}^{k}(t) = \begin{cases} \dfrac{[\square_{ij}(t)]^{\alpha}.[\square_{ij}]^{\beta}}{\sum_{k=n}[\square_{ij}(t)]^{\alpha}.[\square_{ij}]^{\beta}} & \text{if } j \in n \\ 0 & \text{Otherwise} \end{cases} \qquad (4)$$

[7]

Where n stands for the cities for the *kth* ant in iteration *t* and *η*ij is computed according to (5):

$$\square_{ij} = \frac{1}{d_{ij}} \qquad (5)$$

[12]

where *dij* is distance connecting cities *i* and *j*.

B. *Simulated Annealing Algorithm*

Simulated annealing was a method initiated by Scott Kirkpatrick, Daniel Gelatt and Mario Vecchi in 1983. SA is originated from process of solid annealing and cooling [15]. The fundamental scheme is to accept moves resultant in solution of worse class than the current solution so as to run away from the local minima. The chance of accepting that the move decreases in search through the parameter temperature. SA method starts by initial solution x, and candidate solution y is then produced (either at random or using any pre-specified rule) from the neighbor-hood of x. The Metropolis-acceptance criterion that models thermodynamic system which moves from one to another state so that the energy is being minimized is used for deciding whether accept y or not [11]. To apply SA algorithm to any specific problem, always specify the neighbor-hood structure and cooling schedule [16]. These options and their corresponding parameter setting have significant impact lying on SA's performance.

1) *Acceptance Probability Function (P)* determines possibility of moving to a more costly state represented in *(6):*

n = number of the cities or points

T_0 = Initial Temperature

T_k = Temperature at the kth instance of accepting new solution state

$T_{k+1} = \beta T_k$, where β is some constant between 0 and 1

$$P(\delta, Tk) = \begin{cases} e^{-\delta/T}k \;,\; \delta > 0 \\ \qquad \text{for } Tk > 0 \\ 1 \;, \qquad \delta \le 0 \end{cases} \qquad (6)\ [16]$$

SA operators are utilized to select the best one from neighbors. Specifically, two positions i and j are selected, which employs operator to produce neighbor solutions. And the best one is used as the candidate solution. The three SA operators which are swap insert and inverse operators are defined as follows [17].

a) Swap operator

swap (a, i, j) means to swap the city in location j and city in location i. The swap(π, i, j)operator will generate new solution A such that A(i) = a(j)and A(j) = a(i). In general, four edges will replaced by swap operator.

b) Insert operator

insert(a, i, j) means to travel city in location j to location i.The insert(a, i, j) operator will generate new solution A such that A(i) = a(j), A(i + 1) = a(i), . . ., A(j) = a(j − 1), in the case of $i < j$, or A(j) =A(j+1), . . ., A(i−1) = a(i), A(i) = a(j), in the case of $i > j$. In general, three edges will replaced by insert operator.

c) Inverse operator

inverse (a, i, j) means to inverse the cities between location i and location j. The inverse(a, i, j) will generate new solution A such that A(i)=a(j), A(i+1) = a(j-1), ….., A(j)=a(i), where $1 \le i,j \le n \wedge 1 \le j-i < n-1$; in addition, if $j - i = n-1$, it means i = 1 and j = n, and then A(i) = a(j) and A(j) = a(i). Two edges will changed by the inverse operator for symmetric-TSP problems.

C. ***The Proposed Algorithm***

This algorithm is proposed by the combination of ACO and SA. The main purpose to design the new method is to raise the convergence rate of ACO. As already known ACO is good for finding global solution but it get trapped in local-minima. To overcome that problem SA is used which is most suitable algorithm for finding the local solution. The main flow of hybrid ACO-SA algorithm is shown in below pseudo code.

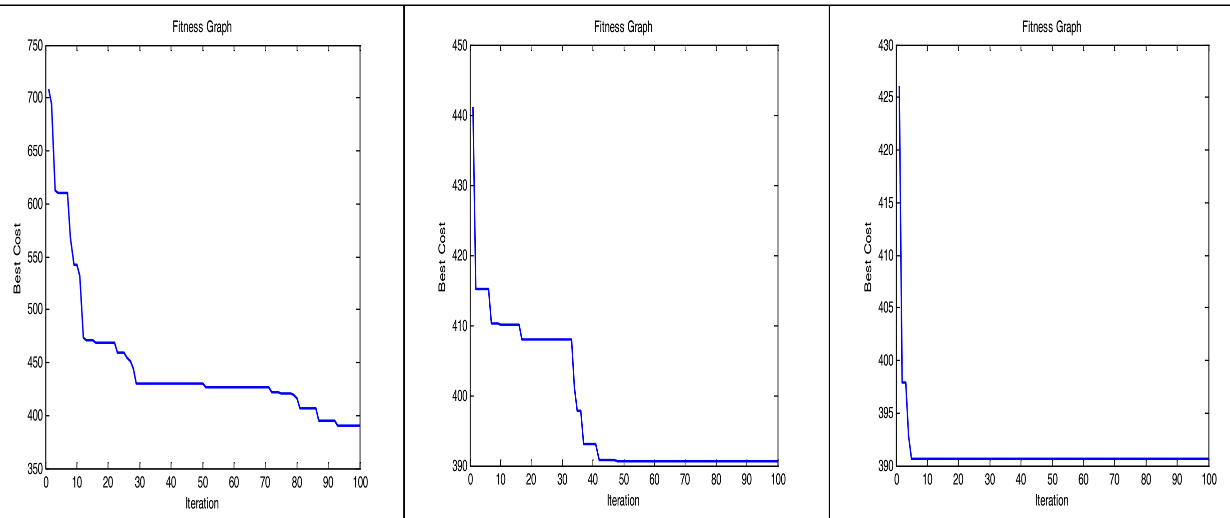
The description of hybrid ACO-SA algorithm is as similar as the workings of ACO algorithm the change occurs in its tour generation step. Firstly, the random solutions are created for m ants. Then the tour is constructed. For constructing best tour firstly ACO construct the tour which selects only high probability tour. After that SA also construct the tour by using its three operators as discussed above. Then, a comparison occurs between both tours and a best one is selected which can solve TSP more rapidly. This is done for finding the solution length. After that the cost is calculated and pheromone is evaluated. At last, the trace is updated and the global best solution is achieved. The cost is calculated and the pheromone or trace is updating. Pseudo-code for ACO-SA

```
Initialize all parameters
for i= 1 to Max-iteration
 for j=1 to m (number of ants)
  Generate probabilistic solutions by using (4)
End
 for k= 1 to n-1( n is solution length)
  Construct ant_tour tour[i]
  Construct tour by SA operators stour[i]
   if(tour[i] > stour[i])
    tour[i] = stour[i]
   end
 end
Calculate the cost
Update the trace by using (2)
Update global best
End
```

V. SIMULATION RESULTS

A typical TSP problem is selected from the general TSPLIB. The simulation software MATLAB 7.9 is used to accomplish the simulation experiments. The ACO algorithm, SA algorithm and hybrid ACO-SA algorithm are used to solve TSP. The parameters of the ACO-SA algorithm are initialized as: α =5, β =4, e =0.15 and e1 =0.97. The total iteration cycles are of 100. Number of cities is equivalent to number of ants

in these algorithms. Seen from Figure 4, firstly all three algorithms simulations results are represented individually. Three graphs are there in which represents the x-axis for number of iterations it takes to give best solution and y-axis represents the best cost as shown in Fig. 2.



Standard SA find optimal solution in 96th iteration | Standard ACO find optimal solution in 43th iteration | ACO-SA find optimal solution in 7th iteration

Figure 2. Simulation Performance of SA, ACO and ACO-SA

The simulation results are revealed in Table 1 and Fig. 3. In term of the solution quality, the proposed ACO-SA algorithm obtains best solution with minimum number of cycles than the standard ACO algorithm and standard SA significantly, which indicates effectiveness of proposed algorithm.

Table I. Comparison of experimental results between ACO , SA and Hybrid ACO-SA

Algorithm Used	***Average Solution***	***Minimum number of cycles to achieve best solution***
Standard SA	408.33	96
Standard ACO	390.68	43
Hybrid ACO-SA	390.68	7

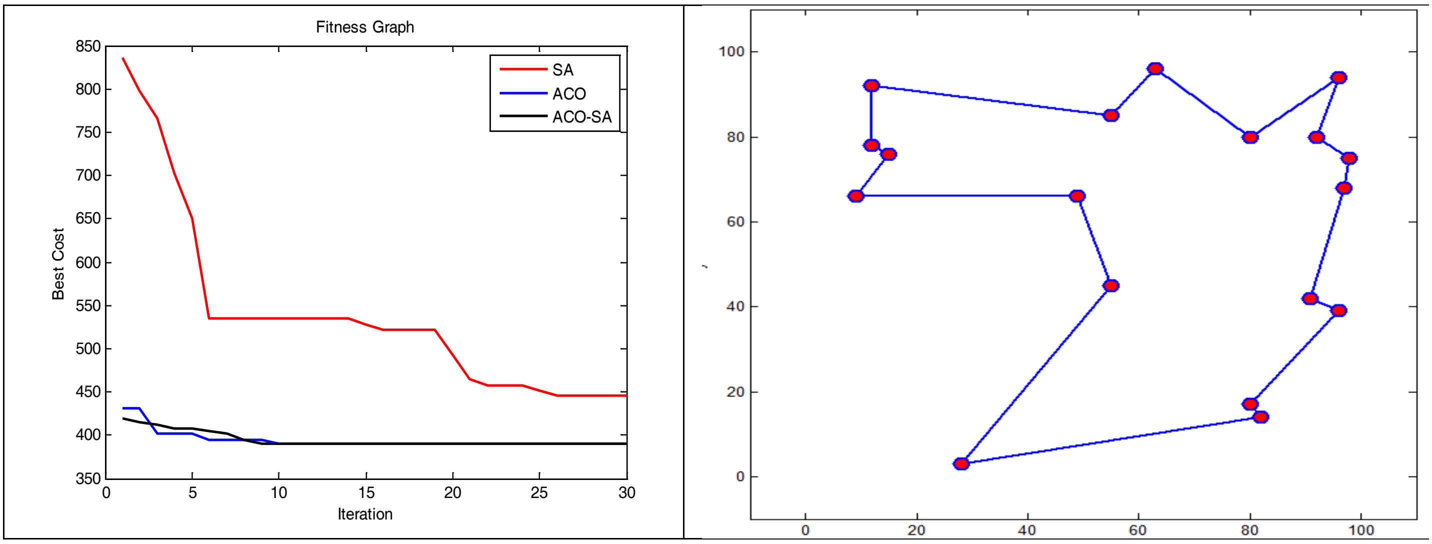


Figure 3. Convergence curves under the proposed ACO-SA algorithm

VI. CONCLUSION

A new hybridized meta-heuristic algorithm has been introduced called ACO-SA algorithm for TSP. In which the best tour is selected from the ACO algorithm or the SA algorithm. The convergence-rate of proposed algorithm is improved by using SA operators to evade trapping in the local-minima points. Experiments were occurred using data sets taken from the TSPLIB and experimental results of proposed algorithm were compared with Standard SA and Standard ACO algorithms. The results illustrate that proposed algorithm converges to global best solution quickly and accelerate the convergence rate. For future work, more parameters will be included to solve TSP. In addition, further evaluation of performance of proposed hybrid ACO-SA algorithm can be done using asymmetric TSP.

VII. REFERENCES

[1] C. Ye, Z. Yang, T. Yan , "An Efficient and Scalable Algorithm for the Traveling Salesman Problem", Proc.5th IEEE International Conference on Software Engineering and Service Science, 2014, pp. 335-339.

[2] K. H. Hingrajiya, R. Gupta and G. Chandel, "An Approach for Solving Multiple TSP using Ant Colony Optimization", Computer Engineering and Intelligent Systems, Vol.6(2), 2015, pp. 13-17.

[3] Z. A. Aziz, "Ant Colony Hyper-heuristics for Travelling Salesman Problem", Proc. IEEE Symposium on Robotics and Intelligent Sensors (IRIS), Vol.76, 2015, pp.534 – 538.

[4] N. Sathya and A. Muthukumaravel, "A Review of the Optimization Algorithms on Traveling Salesman Problem", Indian Journal of Science and Technology, Vol 8(29), 2015, pp. 1-4.

[5] K. Ilavarasi, and S. Joseph, "Variants of Travelling Salesman Problem: A Survey", Proc. IEEE Conference on Information Communication and Embedded Systems (ICICES), 2014, pp. 1-7.

[6] X. Yang, J. Wang, "Application of Improved Ant Colony Optimization Algorithm on Traveling Salesman Problem", Proc. 28th Chinese Control and Decision Conference (CCDC), 2016, pp.2156-2160.

[7] Z. Hlaing and M. Khine, "Solving Traveling Salesman Problem by Using Improved Ant Colony Optimization Algorithm", International Journal of Information and Education Technology, Vol. 1(5), 2011, pp. 404-409.

[8] W. Zhao, X. Cai and Y. Lan, "A new Ant Colony Algorithm for solving Traveling Salesman Problem", Proc. IEEE Conference on Computer Science and Electronics Engineering ,2012, pp. 530-533.

[9] Z. SWiatnicki, "Application of Ant Colony Optimization Algorithms for Transportation Problems Using the Example of the TSP", Proc. 4th IEEE International Conference on Advanced logistics and Transport (ICAlT), 2015, pp. 82-87.

[10] Colony Optimization to Handle Large-Scale Problems", Swarm and Evolutionary Computation (Elsevier), Vol. 32 , 2017, pp.140-150.

[11] A. M. Mohsen, "Annealing Ant Colony Optimization with Mutation Operator for Solving TSP", Computational Intelligence and Neuroscience, Hindawi, 2016, pp.1-13.

[12] S. Kefi, N. Rokbani, P. Krömer and A. M. Alimi, "Ant Supervised by PSO and 2-Opt algorithm, AS-PSO-2Opt, Applied to Traveling Salesman Problem", Proc. IEEE Conference on Systems, Man, and Cybernetics, 2016, pp.4866-4871.

[13] L. Tao, P. Shi, J. Bai, "Research on Parameter Optimization of ant colony algorithm based on genetic algorithm", Proc. 23rd Springer International Conference on Industrial Engineering and Engineering Management, 2017, pp.131-136.

[14] M. Mahia, O. Baykanb and H. Kodazb, "A new hybrid method based on Particle Swarm Optimization, Ant Colony Optimization and 3-Opt algorithms for Traveling Salesman Problem", Applied Soft Computing Elsevier, Vol.30, 2015, pp.484–490.

[15] X. Zhihong, SongBo, G. Yanyan, "Using Simulated Annealing and Ant Colony Hybrid Algorithm to Solve Traveling Salesman Problem", Proc. Second IEEE Conference on Intelligent Networks and Intelligent Systems, 2009, pp.507-510.

[16] C. Wang, M. Lin and Y. Zhong, "Solving travelling salesman problem using multiagent simulated annealing algorithm with instance-based sampling", International Journal of Computing Science and Mathematics (Inderscience), Vol. 6(4), 2015, pp.336-353.

[17] S. Zhan, J. Lin, Z. Zhang and Y. Zhong, "List-Based Simulated Annealing Algorithm for Traveling Salesman Problem", Computational Intelligence and Neuroscience (Hindawi), 2016, pp.1-12.