International Journal of Advanced Computer Research (ISSN (print): 2249-7277 ISSN (online): 2277-7970)
Volume-4 Number-3 Issue-16 September-2014

A Comparative Study of Geometric Hopfield Network and Ant Colony Algorithm to Solve Travelling Salesperson Problem

Yogeesha C.B[1], Ramachandra V Pujeri[2]

Abstract

The classical methods have limited scope in practical applications as some of them involve objective functions which are not continuous and/or differentiable. Evolutionary Computation is a subfield of artificial intelligence that involves combinatorial optimization problems. Travelling Salesperson Problem (TSP), which considered being a classic example for Combinatorial Optimization problem. It is said to be NP-Complete problem that cannot be solved conventionally particularly when number of cities increase. So Evolutionary techniques is the feasible solution to such problem. This paper explores an evolutionary technique: Geometric Hopfield Neural Network model to solve Travelling Salesperson Problem. Paper also achieves the results of Geometric TSP and compares the result with one of the existing widely used nature inspired heuristic approach Ant Colony Optimization Algorithms (ACA/ACO) to solve Travelling Salesperson Problem.

Keywords

Hopfield Neural Networks, Combinatorial Optimization Problem, Geometric –TSP, Ant Colony Algorithm - ACA.

1. Introduction

The Travelling Salesperson Problem [1] is a classic optimization problem that defines easy solution. The TSP problem is NP-complete problem. The conventional approach of comparing the cost function for alternate solutions and picking the most optimum, fails in the case of TSP because of the enormously large number of alternate solutions that need to be examined. The TSP problem is defined as there is a list of cities that are to be visited by salesperson. A salesperson starts from a city and come back to the same city after visiting all the cities. Here the objective is to find shortest path, which follows following constrains:

1. The total length of the loop should be a minimum.
2. The salesperson cannot be at two different places at the same time.
3. The salesperson should visit each city only once.
4. The salesperson should visit each city once and only once.

In temporal processing, we can stimulate a Multilayered Feed Forward Networks as a dynamic mapper by using a memory structures, and important way in which time can be built into the operation of a neural network in an implicit manner is through the use of feedback. There are two basic ways of applying feedback to a neural network: local feedback at the level of a single neuron inside the network, and global feedback encompassing the whole network. Local feedback is a relatively simple matter to deal with, but global feedback has much more profound implications. In the neural network literature, neural network with one or more feedback loops are referred to as recurrent networks.

In the general Hopfield network [2] case, neural networks consist of a (often very high) number of neurons, each of which has a number of inputs, which are mapped via a relatively simple function to its output. Networks differ in the way their neurons are interconnected (topology), in the way the output of a neuron determined out of its inputs (propagation function) and in their temporal behavior (synchronous, asynchronous or continuous) [3].

Ant colony algorithm (ACA) constitutes some heuristic optimizations. Initially proposed by Macro Dorigo [4] in 1992 in his PhD thesis “Optimization, learning, and Natural Algorithms”, the first algorithm was aiming to search for an optimal path in a graph, based on the behavior of ants seeking a path between their colony and a source of food. The original idea has since diversified to solve a wider class of numerical problems, and as a result, several problems

Manuscript received August 30, 2014.
Yogeesha C.B, Department of Computer Science and Engineering, Karpagam University, Coimbatore, TN, India.
Ramachandra V Pujeri, Department of Computer Engineering, SKN Sinhgad Institute of Technology and Science, Lonavala, Pune, Maharashtra, India.

have emerged, drawing on various aspects of the behavior of ants.

The final section of the paper will compare result obtained from new approach Geometric Hopfield Network with existing Ant Colony Algorithm to solve TSP problem.

2. Geometric Hopfield Network to solve TSP

In a Hopfield neural network [5] [6] each processing element has an external input and has weighted connections from other neurons. The continuous Hopfield model is used to find solutions for 10 city problem. The 10 co-ordinates used as an input to the problem. Like CityA(x1,y1), CityB(x2,y2), ...CityJ(x10,y10). The distances between the cities are calculated using discrete Euclidean length:

$$Dist_{x,y} = Co-ordinatres_{x1,y1,x2,y2} = \sqrt{(x1-x2)^2 + (y1-y2)^2}$$

Thus the number of paths that need to be examined are given by:

$$N = \frac{n!}{2n} = \frac{(n-1)!}{2}$$

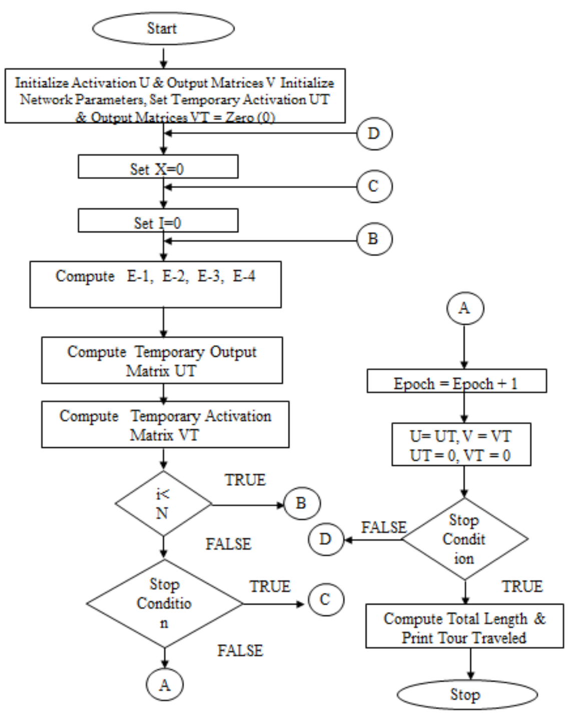

Figure 1: Geometric-TSP Flow chart

The basic steps for solving the geometric TSP using a continuous Hopfield model is shown figure 1.

The output function used is:

$$G(U_{x,i}) = OutputV_{x,i} = 0.5 \cdot (1 + tanh(\alpha * UT[x,i]))$$

Figure 2 and Figure 3 are describes representation of the Hopfield Network topology for TSP.

	C1	C2	C3	C4	C5
C1	0	0	1	0	0
C2	1	0	0	0	0
C3	0	0	0	0	1
C4	0	1	0	0	0
C5	0	0	0	1	0

Figure 2: Cost Tour matrix of 5-City problem

(i,j) (k,l)	1	2	3	4	5	
1			$V_{(1,2)}$			**A**
2	$V_{(2,1)}$					**B**
3					$V_{(2,5)}$	**C**
4		$V_{(4,2)}$				**D**
5				$V_{(1,4)}$		**E**

Figure 3: Tour matrix obtained as the output of the network

By using below equation calculate the distance matrix for the given co-ordinates:

$$Dist_{x,y} = Co-ordinatres_{x1,y1,x2,y2} = \sqrt{(x1-x2)^2 + (y1-y2)^2}$$

Initialize Activation Matrix U and Output Matrix V Activation Matrix:

$$V_{i,j} = \sum_i \sum_j \frac{Cities}{100} + RandomNoise$$

$$RandomNoise = \frac{\left(\frac{RAND}{RANDMAX}\right)}{10.0} - 0.5$$

where random noise is between –0.05 to +0.05

Output Matrix:

$$U_{i,j} = \sum_i \sum_j atanh\left(2 \cdot \frac{V_{i,j-1}}{\alpha}\right)$$

V = Activation Matrix
i = a specific row
j = a specific column

$$E_1 = \sum_x \sum_i \sum_{(j \neq i)} V_{x,i}$$

V = Activation Matrix
x = a specific row (xth city)
i = ith neuron in xth row
j = any neuron other than i in xth row

$$E_2 = \sum_x \sum_i \sum_{(y \neq x)} V_{y,i}$$

V = Activation Matrix
x = a specific column
i = ith neuron in xth column
y = any neuron other than i in the xth column

$$E_3 = \sum_x \sum_i \sum_j \sum_k V_{j,k}$$

V = Activation Matrix
x = a specific row
i = a specific column
j = any neuron in xth row
k = any neuron in ith column

$$E_4 = \sum_x \sum_i \sum_{(y \neq x)} Dist_{x,y} \cdot \left(V_{y,i-1} + V_{y,i+1}\right)$$

Dist = Distance Matrix
V = Activation Matrix
x = a specific row
y = any row other than the xth row
i = any neuron in yth row (left to y or right to y)

$$\Delta U = \Delta T \cdot \left(-1.0 \cdot U_{x,i} - A \cdot E_1 - B \cdot E_2 + C \cdot \left(N - E_3\right) - D \cdot E_4\right)$$

UT and VT are temporary Activation Matrix and Output Matrix

$$UT_{x,i} = U_{x,i} + \Delta U$$

$$G\left(U_{x,i}\right) = Output V_{x,i} = 0.5 \cdot \left(1 + \tanh(\alpha * UT[x,i]\right)$$

$$VT_{x,i} = G\left(U_{x,i}\right)$$

$$\Delta V = V_{x,i} - VT_{x,i}$$

Values of A, B, C, D, N, alpha (α) and deltat (Δt) are used as network parameters. Program output for corresponding initial settings of random 20 tours are shown in table 1. Table lists Run Trails, length, Epoch (integration) and Time consumed to complete a circuit. Table 2 shows the best, worst and average of top 3 optimal results.

Table 1: G-TSP approach for 10 cities, random 20 tours run

Trail	Length	Epoch	Time (Sec)
1	4.481890	269	0.050000
2	3.316609	690	0.130000
3	2.986585	373	0.070000
4	4.162376	683	0.120000
5	4.698537	1633	0.300000
6	3.610237	103	0.020000
7	3.271382	134	0.030000
8	3.763221	849	0.150000
9	3.942118	175	0.030000
10	3.385619	115	0.030000
11	3.646618	827	0.160000
12	3.620115	1300	0.240000
13	3.918091	158	0.020000
14	2.947796	127	0.020000
15	4.284040	166	0.030000
16	4.070274	454	0.090000
17	4.056540	266	0.060000
18	3.005290	158	0.040000
19	3.990957	146	0.030000
20	3.712748	437	0.080000

Table 2: G-TSP for 10 cities with 3 optimal results

Best Tour for 10 cities	Epoch	Time (sec)
Best Tour	127(2.947796)	0.020000
Average Tour	158 (3.005290)	0.040000
Worst Tour	373(2.986585)	0.070000

Table 3: Best result of G-TSP for 12, 11 and 10 city problem

City	Length	Epoch	Time (Sec)
12	4.604446	453	1.9899
11	4.205311	348	0.5182
10	2.947796	127	0.02000

Above table 3, shows the results of G-TSP problem for cities 12, 11 and 10. These are the best optimal results chosen from multiple run. Table describes the time taken is gradually get increases when the number of cities are getting added to problem. For TSP best and worst time complexity falls between $O(n)^2$ and O(n!) respectively. Here we can observe that the time taken to visit all the cities with minimum length is very close to $O(n)^2$.

3. Ant Colony Algorithms (ACA) to solve TSP

A great deal of work is already done by many researches on solving Travelling Salesperson Problem using Ant Colony Algorithms. For comparative study of Geometric Hopfield TSP, we have chosen an existing technical paper "A Comparative Study of ACO, GA and SA for Solving Salesman Problem" [7]. This paper explores three widely used nature inspired heuristic approaches Ant Colony Optimization (ACO), Genetic Algorithms (GA) and Simulated Annealing (SA) to solve TSP and it compare results of ACO, GA and SA for standard TSPLIB [8] instances.

Procedure ACA
- Step 0. Begin ACA procedure
- Step 1. Generate pheromone trails and other parameter
- Step 2. Check for termination criteria (meet or not meet)
- Step 3. If meet go to Step 7
- Step 4. Construct solutions
- Step 5. Update pheromone trails
- Step 6. Go to Step 2
- Step 7. Post process result and output
- Step 8. End ACA procedure

Key parameters used are distances between two ants, pheromone update which includes pheromone deposit, and pheromone evaporation.

Initially ants start their search roams randomly. An ant selects the next node to be visited by probabilistic equation. When ant k is on node i, the probability of going to node j is given by equation as follows [9]:

$$P_{ij}^{k} = (\tau_{ij})^{\alpha} (\eta_{ij})^{\beta} / \sum_{l \in N_l^k} (\tau_{ij})^{\alpha} (\eta_{ij})^{\beta} \quad \text{if } j \in_i^k N$$

Where,
N_l^k = adjacent node of k
α = local pheromone coefficient
β = heuristic coefficient
η = 1/d which is the inverse od distance between i & j

N_l^k is which still not visited by ant i. α controls the amount of contribution pheromone plays in a components probability of selection and is commonly set to 0.1. β which controls the amount of contribution problem specific heuristic information plays in a components probability of selection and is commonly between 2 and 5, such as 2.5. The total number of ants (m) is commonly set low, such as 10. The arcs which is used by the most ants and which is the shortest, receives more pheromone and will be used by the ants in future.

Function pheromone update which consists of pheromone deposit and pheromone evaporation. Pheromone values are updated each time an ant travels from one node to another. A first Pheromone value on each arc is decreased by constant factor which is known as pheromone evaporation. Then some amount of pheromone is added to each node which is being traversed by each ant, is known as pheromone deposit. Pheromone evaporation is given by equation follows:

$$\tau_{ij} \leftarrow (1-\rho)\,\tau_{ij}$$

Where, ρ is the evaporation rate.

Each ant drops some amount of pheromone on each node which is known as pheromone deposit and given by equation follows:

$$\tau_{ij} \leftarrow \tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k}$$

Where,
m = number of ants
$\Delta\tau_{ij}^{k}$ = amount of pheromone drop on k node and is calculated as:

$$\Delta\tau_{ij}^{k} = \{1/C^{k} \text{ if arc(i,j) belongs to } \tau^{k}\text{, otherwise } 0$$

Where, C^k is the length of the tour by the k^{th} ant.

Table 4: ACA approach result to solve TSP for different number of cities

Cities (no)	Best	Worst	Average
10	60.50	63.81	61.84
29	9999.20	11151.22	10440.04
51	529.15	578.12	558.04
16	74.00	82.64	78.08
30	452.96	510.01	479.57
48	39930.58	42895.88	41374.88

In experimental result for ACA to solve standard TSP instances are downloaded from TSPLIB [8]. Table 4 lists Number of Cities and length of the optimal tour for ACA. The results given here are best, worst and average of fifteen runs.

4. Comparison of using Hopfield Network and ACA

The advantages of using Hopfield network setup are very optimal for the solution of combinatorial optimization problem. It can be easily used for the optimization problems like that of TSP. It gives very accurate result due to very powerful and complete Energy equation. This neural network approach is very fast compared to standard programming techniques used for TSP solution. With very few changes this algorithm can be modified to get the approximate solution for many other NP-complete problems. Problem faced with this approach understand of Energy, output, activation and weight update functions are very difficult. The setting for various parameter values like A, B, C, D, NN, α, and Δt was a challenge. The best value was chosen by trial and error. Improvement is still possible for this parameters value. Many times the algorithm converged to local minima instead of global minimum. This problem was mostly resolved by adding a random noise to the initial inputs of the system. The testing of algorithm gets difficult as the number of cities increase. So testing was carried out till 12 cities problem.

ACA to solve TSP has the advantage of distributed computing. It is robust and also easy to accommodate with other algorithms. ACA have advantage over simulated annealing and Genetic Algorithms (GA) approaches of similar problems (such as TSP) when the graph may change dynamically, the ant colony algorithms can be run continuously and adapt to changes in real time. On the other side limitation of this approach is; ACA can solve some optimization problems successfully, but we cannot prove its convergence. It is prone to failing in the local optimal solution, because the ACA updates the pheromone according to the current best path.

The best and worst time complexity of TSP which falls between $O(n)^2$ and $O(n!)$.

From table 3, it shows that with Hopfield network approach to visit all the cities with minimum length is very close to $O(n)^2$. Table 4 lists TSP results of number of cities and lengths of the optimal tour for ACA. TSP instance provides some cities with their co-ordinates. The results given here are best, worst and average of 15 runs. If there is any change in city co-ordinates and addition of cities to this approach, the complexity of problem will increases by a factorial factor and also it shows an increase in best, worst average instances.

5. Conclusion and Future work

This paper presents a study of comparative view of two of the widely used optimization solving techniques namely Geometric Hopfield Network and Ant Colony Algorithms. Geometric Hopfield Network is a process used by multilayer neural network to map via a relatively simple function to its output and Ant Colony Algorithm is the process used by ants to forage food source. They use pheromone trail deposition technique to map their way.

The sections 2 and 3 shows the experimental results obtained on standard TSP problem. In section 2, the Geometric Hopfield Network – TSP shows that the best time required solving 12, 11 and 10 cities problem is very close to $O(n)^2$ i.e. result is obtained only with 453, 348 and 127 epochs respectively. In section 3, Ant Colony Algorithm - TSP shows best result when number of cities are less. If a single city is added to this problem, complexity of problem will increases by a factorial factor. So in case of ACA approach the result shows an increase in best, worst average instances.

There is lot of scope for research work in the field of Artificial Intelligence, Evolutionary Computation, Evolutionary Techniques, Neural Networks, Ant Colony Optimization and Combinatorial Optimization. By combining Artificial Neural network and Ant Colony Algorithm techniques or combining two or more techniques one can complement each other and validate respective limitations.

Acknowledgment

Authors wish to thank each and every one who have helped and supported to complete this research work.

References

[1] Byung-In Kim, Jae-Ik Shim, Min Zhang, "Comparison of TSP Algorithms" Project for Models in Facilities Planning and Materials Handling, December 1998.
[2] Dan W. Patterson, "Artificial Neural Networks: Theory and Applications", Prentice Hall of India private Limited, pp. 14-19, 141-178, 1996.
[3] Elaine Rich and Kevin Knight, "Artificial Intelligence", Second Edition, Tata McGraw-Hill Publishing Company Limited, pp. 40-62, 1991.
[4] M. Dorigo, L. Gambardella, "Ant colonies for the Traveling salesman problem." Biosystems 43, pp. 73-81, 1997.
[5] Yogeesha CB and Dr. Ramachandra V Pujeri, "Randomized Algorithms: On the Improvement of Searching Techniques Using Probabilistic Data Structure Linear Linked Skip Lists", http://link.springer.com/chapter/10.1007/978-81-322-0740-5_19, International Conference on Advances in Computing, pp 147-153, 1996.
[6] Kate Smith, Marimuthu Palaniswami and Mohan Krishnamoorthy, "Neural Techniques for Combinatorial Optimization with Applications", IEEE Transactions on Neural Networks, VOL. 9, NO. 6, pp. 1301-1318, November 1998.
[7] Sharad N. Kumbharana and Prof. Gopal M. Pandey, "A Comparative Study of ACO, GA and SA for Solving Travelling Salesman Problem", International Journal of Societal Applications of Computer Science, Vol 2 Issue 2, pp. 224-228, February 2013.
[8] TSPLIB95: Ruprecht - Karls - Universitat Heildelberg "Discrete and Combinatorial Optimization" http://www.iwr.uiniheidelberg.de/groups/comopt/software/TSPLIB95/, 2011.
[9] M. Dorigo, T. Stutzle, "Ant Colony optimization", A Bradford book, MIT Press Cambridge, Massachusetts London, England, 2004.

Yogeesha C.B. Born in Chikmagalur, Karnataka state in India, in 1977. He is a Ph.D Research Scholar (July – 2011 batch) from Department of Computer Science and Engineering, Karpagam University in Coimbatore. Currently he is working as Sr. Software Engineer in Cisco Systems India Pvt Ltd, Bangalore from 2006. He obtained his master (MTech) in Computer Engineering from National Institute of Technology, Surathkal, Karnataka in 2004. He received graduation (BE) in Computer Science and Engineering from Kuvempu University, Adichunchanagiri Institute of Technology, Chikmagalur in 1999. His research areas includes: Advance Algorithms, Artificial Intelligence, Virtualization, Contact Center, Agile Methodology, Software Engineering, Software Testing and Automation.

Dr. Ramachandra V Pujeri. Born in Bijapur, Karnataka state in India, in 1973, received the B E in Electronics and Communication Engineering from Karnataka University, Dharwad, ME in Computer Science and Engg from PSG College of Technology, Coimbatore, Ph.D. in Information and Communication Engineering from Anna University, Chennai, MBA in Human Resource Management, from Pondicherry University, Pondicherry, in 1996, 2002, 2007 and 2008 respectively. He is active life member of ISTE, SSI, MIE, ACS and IEE. He has written three textbooks. He is having around 20 years of teaching experience in the various top ten engineering colleges in India. He is an active expert committee member of AICTE, NBA, DoEACC, NACC and various Universities in India. Currently, under him ten research scholars pursuing their Ph.D. His research interests lie in the areas of Computer Networking, Operating System, Software Engineering, Software Reliability, Modelling and Simulation, Quality of Services and Data Mining. Currently, he is working as Vice Principal and Head of Computer Engineering Department in SKN Sinhgad Institute of Technology and Science, Lonavala, Pune, Maharashtra.