A New Hybrid Algorithm for Traveler Salesman Problem based on Genetic Algorithms and Artificial Neural Networks

Alireza Arab Asadi
Mathematics and Computer Faculty, Shahid Bahonar University, Kerman, Iran

Ali Naserasadi
Industrial and Mining Faculty, Shahid Bahonar University, Kerman, Iran

Zeinab Arab Asadi
Computer Department, Amirkabir University, Tehran, Iran

ABSTRACT
Traveler Salesman Problem (TSP) is one the most famous and important problems in the field of operation research and optimization. This problem is a NP-Hard problem and it is aimed to find a minimum Hamiltonian cycle in a connected and weighed graph. In the last decades, many innovative algorithms have been presented to solve this problem but most of them are inappropriate and inefficient and have high complexity. In this paper, we combined Hopfield neural network with genetic algorithm to solve this problem, and showed that the results of the algorithm are more efficient that the other similar algorithms.

General Terms
Algorithms

Keywords
Travelers Salesman Problem, Genetic Algorithm, Hopfield Neural Network, NP-Hard Problem

1. INTRODUCTION
We have a weighed and connected graph with *n* vertices (cities). We want to start a tour from one these vertices, visit all of the other vertices for exactly one time and finally return to start vertex. This tour is called Hamiltonian tour. In the TSP, we want to find a minimum cost Hamiltonian cycle. Finding the minimum Hamiltonian cycle is easy with low number of vertices. But with increasing of the number of vertices (cities), the problem got harder and becomes a NP-Hard problem. This problem was introduced in the 18th century for first time by William Hamilton and Thomas Kirkman and in the 1930s was published in public form by mathematician such as Karl Menger from Harvard and Hassler Whitney from Princeton [1, 2].

Many algorithms were introduced to solve this problem. Most of them are trying to find a solution near to optimal tour. Freisienen and et al. used heuristic methods to solve the problem [3]. Laarhoven and et al. Used Simulated Annealing [4]. Dorigo and et al. used ant colony algorithms [5]. Goldberg and et al. used Genetic Algorithms [6] and Al Rahedi defined new parameter for Genetic Algorithm [7]. Hejazi combine ant colony algorithm with Genetic Algorithms [8]. Also, Feng and et al. used Hopfield neural networks to solve this problem [9]. Others used Cohnen neural networks to solve this problem [10].

In this paper, first we employed Hopfield neural networks and then used its results as initial population of genetic algorithm. Finally we show that the results of this hybrid algorithm are better than similar and simple algorithms.

2. HOPFIELD NEURAL NETWORK
Hopfield neural network is a recursive and monolayer neural network with a symmetric weight matrix that its diagonal elements are equal to zero. This network can be continuous or discrete with time and is a fully connected network. We can use this network with a well defined energy function to find an answer near to optimal one [11]. In the TSP, we define n^2 neurons for n cities. Also, we assume that the graph is a full mesh. So, there is an edge between each pair of vertices. The tour is showed by a matrix, so, the element in row i and column j shows the ith city in the jth situation. Figure 1 shows a simple example for a Hamiltonian tour in a TSP with 4 cities. The tour is $D \rightarrow A \rightarrow C \rightarrow B \rightarrow D$.

	1	2	3	4
A	0	1	0	0
B	0	0	0	1
C	0	0	1	0
D	1	0	0	0

Figure 1.A simple example for a Hamiltonian Tour in a TSP with 4 cities

TSP is an optimization problem that has some constraints. The first constraint is that in optimal tour each vertex (city) should be visited exactly one time. In other word, in each row of matrix we can have exactly one element with value of one. We show this constraint as relation 1.

$$\sum_i \sum_k \sum_{j \neq k} x_{ik} x_{ij} \quad (1)$$

If ith city exist on kth situation, the value will be equal to one and else it will be equal to zero. The second constraint is that in each level of tour only one city should be visited, so in each column we just have an element with value one. This constraint is showed via relation 2.

$$\sum_i \sum_k \sum_{j \neq k} x_{ki} x_{ji} \quad (2)$$

The third constraint is that the optimum tour should visits all cities. This constraint is showed via relation 3.

$$[(\sum_i \sum_k x_{ik}) - n]^2 \quad (3)$$

The last constraint is that the value of tour should be minimized. This constraint is showed via relation 4.

$$\sum_k \sum_{j \neq k} \sum_i d_{ij} x_{ki} (x_{j,i+1} + x_{j,i-1}) \quad (4)$$

In relation 4, d_{ij} shows the distance between city i and city j (the weight of edge between i and j). So, we can define the Hopfield network's energy function for TSP as relation 5.

$$E = (A_1 \sum_i \sum_k \sum_{j \neq k} x_{ik} x_{ij}) + (A_2 \sum_i \sum_k \sum_{j \neq k} x_{ki} x_{ji}) + (A_3 [(\sum_i \sum_k x_{ik}) - n]^2 + A_4 d_{ij} x_{ki} (x_{j,i+1} + x_{j,i-1}) \quad (5)$$

In relation 4, values of A_1, A_2, A_3 and A_4 are constant. The minimum value of function E will be achieved when the fourth sentence minimized and the first three sentences equals to zero. Therefore, the constraints will be satisfied and the value of E will be equal to TSP answer [9].

2.1 Weight Matrix in the Network

In the Hopfield network, the weight matrix is a $n^2 \times n^2$ matrix and calculated via relation 6.

$$w_{xi,yj} = -A\delta_{xy}(1 - \delta_{ij}) - B\delta_{ij}(1 - \delta_{xy}) - C - Dd_{xy}(\delta_{j,i+1} + \delta_{j,i-1}) \quad (6)$$

In the above relation (relation 6), the function δ is calculated via relation 7 [9].

$$\delta_{ij} = \begin{cases} 1, & if\ i = j \\ 0, & if\ i \neq j \end{cases} \quad (7)$$

In the Hopfield network for TSP, activation function is defined as relation 8 [9].

$$a_{ij} = \Delta t(T_1 + T_2 + T_3 + T_4 + T_5),$$
$$T_1 = \frac{-a_{ij}}{t}\ ,\ T_2 = -A_1 S_i x_{ik}\ ,\ T_3 = -A_2 S_i x_{ik}\ ,$$
$$T_4 = -A_3(S_i S_k S_{ik} - m)\ ,$$
$$T_5 = -A_4 S_k d_{ik}(x_{k,j+1} + x_{h,j-1}) \quad (8)$$

Finally, the output of Hopfield network obtains from relation 9 [9].

$$x_{ij} = \frac{(1 + \tanh(\lambda a_{ij}))}{2} \quad (9)$$

3. GENETIC ALGORITHM

Genetic algorithm is one of search techniques that are trying to find optimal solution for optimization and operation research problems. This algorithm is a special type of evolutionary algorithms that uses biological techniques such as inheritance and mutation. Genetic algorithm is an optimization technique that uses Darwin natural selection techniques to find the optimal formula. For optimizing problem's target function, this technique tries to generate a new generation of population from current generation. Each population consists from k individual and each individual is represented as a string over a finite alphabet. The algorithm starts with a random (or preprocessed) initial population and uses some special steps to generate a new and better population from this population. To do this, the algorithm uses a fitness function to evaluate the answers. In each round of algorithm, if at least one individual fits enough according to the fitness function, the algorithms stops and returns that individual as the answer [12, 13, and 19].

3.1 Genetic Algorithm Basic Operations

As mentioned earlier, Genetic Algorithm uses three basic operations to generate the new generation of population form current one [14, 15, and 16]. These operations are as follows:

1. Selection: select some of individuals form current population according to fitness function to generate new generation. In this operation, individuals with higher fitness value have more chance for selection.
2. Crossover: in this operation, a new individual is created via two selected individual form current generation. The crossover operation repeated for k times to generate new population.
3. Mutation: this operation can make a random change in an individual to increase its fitness value.

Figure 2 describe the genetic algorithm that implements all above operations for TSP.

```
Function TSP(Initial_Population) returns an individual
 { current = Initial_Population;
   repeat
    { new_generation = ∅;
      For (int i=1; i≤k; i++)
      { first = randomly selected individual from current
                 according to fitness function;
        second = randomly selected individual from current
                  according to fitness function;
        new_ind = Crossover(first, second);
         if (fitness(new_ind)≤Threshold) Mutation(new_ind);
        new_generation = new_generation ∪ new_ind;
      }
    current = new_generation;
    }
  until some individual from current fits enough;
  return (the best individual form current according to fitness);
 }
```

Figure 2.Genetic algorithm for TSP

4. HYBRID ALGORITHM FOR TSP

To solve TSP, innovative and heuristic methods have an important role [17]. In the last decade, many methods in this field were presented [18]. In this section, we present a new hybrid algorithm for TSP based on Hopfield neural network and Genetic algorithm. In this algorithm, we educate the problem via Hopfield network and then use the output of this network (that is near to optimal answer) as initial population of Genetic algorithm. This can help improving of Genetic algorithm efficiency via using an initial population near to actual answer instead of a random initial population. So, we expect better results that the other algorithms. Figure 3 shows the hybrid algorithm steps for TSP. As it shows, first we formulate the TSP. We assume that the associated graph is a full mesh to test the algorithm in the worst case. As mentioned earlier, in the second step we use Hopfield algorithm to educate the problem and obtain an answer near to actual and optimum answer. At last, we use the introduced Genetic algorithm to obtain the optimum solution.

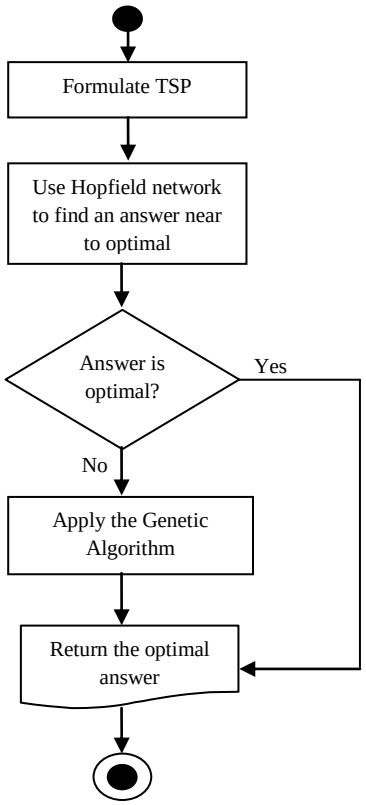

Figure 3.The hybrid algorithm for TSP based on artificial neural network and genetic algorithm

5. IMPLEMENTATION AND RESULTS

To solve TSP, first we defined the structure of Hopfield network. In this network, we had n neurons (based on the number of cities). Also, we initialized the constant values of the network via relation 10.

$$A_1 = 0.5,\ A_2 = 0.5,\ A_3 = 0.2,\ A_4 = 0.5 \qquad (10)$$

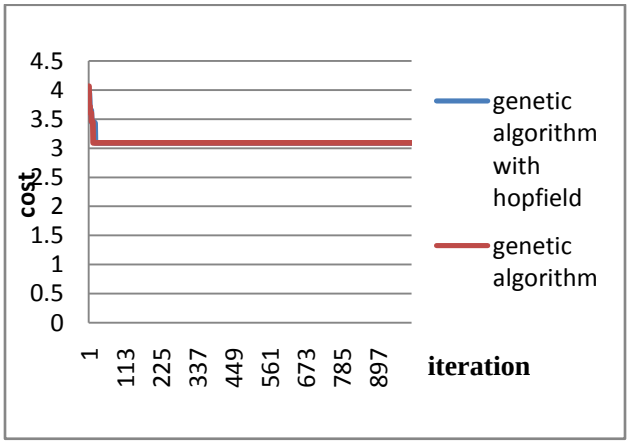

Figure 4.The result of the algorithm on TSP with 10 cities

Also, we initialized the other variables of the network via relation 11.

$$m = 15,\ \tau = 1,\ \lambda = 3 \qquad (11)$$

At last we run the hybrid algorithm on TSPs with 10, 20 and 26 cities. Figures 4, 5 and 6 show the results.

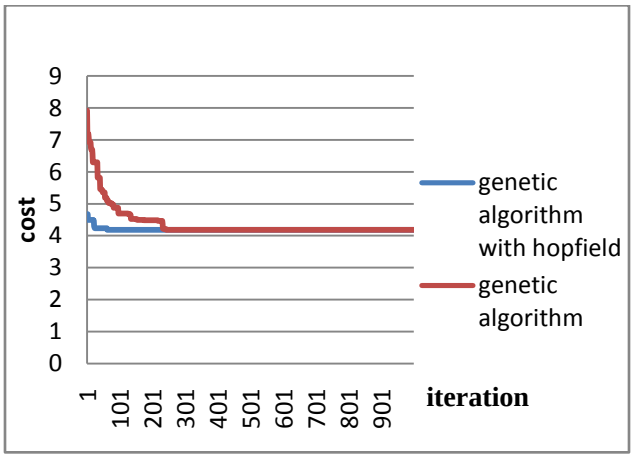

Figure 5.The result of the algorithm on TSP with 20 cities

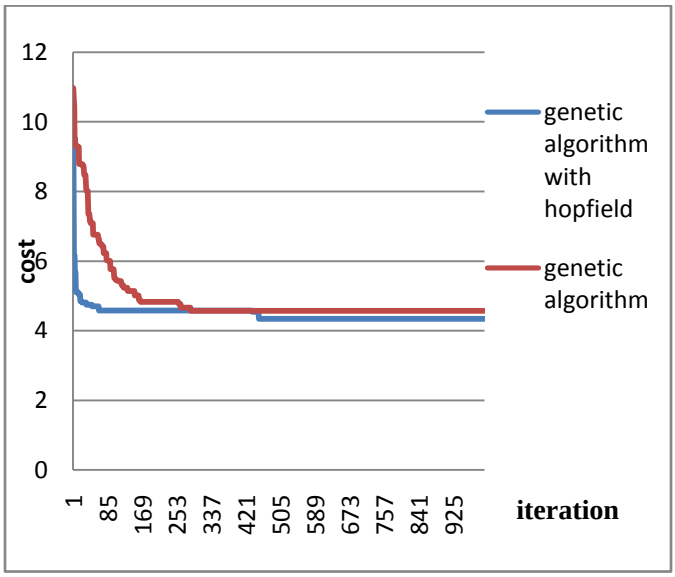

Figure 6.The result of the algorithm on TSP with 26 cities

6. CONCLUSION

Traveler Salesman Problem (TSP) is one the most famous problems in the field of operation research and optimization. This problem is a NP-Hard problem and so, there is not any algorithm with polynomial time complexity for it. Therefore, many heuristic algorithms were presented for this problem such as using simulated annealing, genetic algorithm and artificial neural network. In this paper, we presented a hybrid algorithm for TSP based on combination of artificial neural network and genetic algorithm. Also, we showed that the hybrid algorithm has better time and space complexity than the other algorithms.

7. REFERENCES

[1] T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein, Introduction to Algorithms, 3rd edition, MIT press, 2010.

[2] C. Neapolitoan and K. Naimipour, Foundations of Algorithms Using C++ Pseudocode, Jones and Bartlett Publishers, 2004.

[3] B. Freisieben and P. Merz, New Genetic Local Search Operator for Travelling salesman Problem, Conference on Parallel Problem Solving From nature, app. 890-899, 1996.

[4] P. van Laarhoven and E. H. L. Aarts, Simulated Annealing: Theory and Applications, KluwerAcademic, 1987,

[5] M. Dorigo and L. M. Gambradella, Ant Colony System: A Cooperative Learning Approach to theTravelling Salesman Problem, JEEE Transaction on Evolutionary Computation, Vol. l, pp. 53-66, 1997.

[6] ID.E. Goldberg, Genetic Algorithm in Search, Optimization and Machine Learning. Addison-Wesley, 1989.

[7] Naef Taher Al Rahedi and Jalal Atoum, Solving TSP problem using New Operator in GeneticAlgorithms, American Journal of Applied Sciences 6(8):1586-1590, 2009.

[8] S. R. Hejazi, R. Soltani, Hybrid Algorithm for TSP based on Ant Colony and Genetic Algorithm, 4th conference on industries, 2006.

[9] Gang Feng and Christos Douligeris, Using Hopfield networks to solve traveling salesman problems based on stable state analysis technique, Neural Networks, 2000. IJCNN 2000, Proceedings of the IEEE-INNS-ENNS International Joint Conference.

[10] S.Ghafurian and N. Javadian, Ant Colony Algorithm for Solving fixed destination multi-depot multiple travelling salesman problems, Applied Soft Computing, Vol. 11, Issue 1, January 2011, pp. 1256 – 1262.

[11] D. Kriesel, a Brief Introduction to Neural Networks, http://www.dkiesel.com/en/science/neural_network, 2011.

[12] S. J. Russell and P.Norvig, Artificial Intelligence: A Modern Approach, 3rd edition, Pearson education Inc, 2010.

[13] Banzhaf, Wolfgang; Nordin, Peter; Keller, Robert; Francone, Frank (1998) Genetic Programming – An Introduction, Morgan Kaufmann, San Francisco, CA..

[14] Bies, Robert R; Muldoon, Matthew F; Pollock, Bruce G; Manuck, Steven; Smith, Gwenn and Sale, Mark E (2006). "A Genetic Algorithm-Based, Hybrid Machine Learning Approach to Model Selection". Journal of Pharmacokinetics and Pharmacodynamics (Netherlands: Springer): 196–221.

[15] Mitchell, Melanie, (1996), An Introduction to Genetic Algorithms, MIT Press, Cambridge, MA.

[16] Whitley, D. (1994). A genetic algorithm tutorial. Statistics and Computing 4, 65–85.

[17] Hingston,Philip F.; Barone, Luigi C.; Michalewicz, Zbigniew (2008) Design by Evolution: Advances in Evolutionary Design:297

[18] Eiben,Agoston E.; Smith, James E. (2003) Introduction to Evolutionary Computing

[19] A. Solomon and B. W. Colleti, a Quasibelian landscape of the travelling salesman problem are elementary, Discrete Optimization, Vol. 6, Issue 3, Aug. 2003, pp. 288-291.