Global Journal of Computer Science and Technology
Volume 11 Issue 14 Version 1.0 August 2011
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals Inc. (USA)
Online ISSN: 0975-4172 & Print ISSN: 0975-4350

Application of Neuro-Fuzzy System to Solve Traveling Salesman Problem

By Nitin Jain, Suman Sangwan

Abstract - This paper presents the application of adaptive neuro-fuzzy inference system (ANFIS) in solving the traveling salesman problem. Takagi-Sugeno-Kang neuro-fuzzy architecture model is used for this purpose. TSP, although, simple to describe & mathematically well characterized, is quite difficult to solve. TSP is called a NP-Hard problem, i.e. this problem is as hard as the hardest problem in NP-Complete space. Training of fuzzy system was performed by a hybrid Back-Propagation (BP) and Least-Mean-Square (LMS) algorithm and for optimizing the number of fuzzy rules, subtractive-clustering algorithm was utilized. Then the ANFIS was tested against a number of training data samples. More accurate and quick results were obtained by using ANFIS.

Keywords : *MF, FIS, ANFIS, Subtractive Clustering, BP, LMS.*

GJCST Classification : *I.2.3*

Strictly as per the compliance and regulations of:

Application of Neuro-Fuzzy System to Solve Traveling Salesman Problem

Nitin Jain[α], Suman Sangwan[Ω]

Abstract - This paper presents the application of adaptive neuro-fuzzy inference system (ANFIS) in solving the traveling salesman problem. Takagi-Sugeno-Kang neuro-fuzzy architecture model is used for this purpose. TSP, although, simple to describe & mathematically well characterized, is quite difficult to solve. TSP is called a NP-Hard problem, i.e. this problem is as hard as the hardest problem in NP-Complete space. Training of fuzzy system was performed by a hybrid Back-Propagation (BP) and Least-Mean-Square (LMS) algorithm and for optimizing the number of fuzzy rules, subtractive-clustering algorithm was utilized. Then the ANFIS was tested against a number of training data samples. More accurate and quick results were obtained by using ANFIS.

Index Terms : MF, FIS, ANFIS, Subtractive Clustering, BP, LMS

I. Introduction

The Traveling Salesman Problem (TSP) is one of the oldest combinatorial/optimization problems. TSP can be formulated as: A salesman visits 'n' number of city in such a way that the salesman starts journey by selecting one city among n cities, visits the other cities one by one and returns to the first city from where the journey was started. So the journey of the salesman provides a complete tour that consists of all cities. Now, the resultant of TSP is to find a tour with minimum cost.

TSP is a classic NP-Complete task requiring enormous amount of computational time, irrespective of the algorithm used in that the number of possible tours is extremely large even for problems containing a small number of cities. TSP is a constrained non-linear optimization problem. It is simple to describe, mathematically well characterized. But to find the actual best solution to TSP is computationally very hard, called a NP-complete problem. [7]

The TSP is computationally intensive if an exhaustive search is to be performed comparing all possible routes to find the best one. For an n-city tour, there are n possible paths. Due to degeneracies, the number of distinct solutions is less than n. The term distinct in this case refers to tours with different total distances. For a given tour, it does not matter which of the n cities is the starting point, in terms of the total distance traveled. This degeneracy reduces the number of distinct tours by a factor of n. Similarly, for a given circuit, it does not matter in which of two directions the salesman travels. This fact further reduces the number of distinct tours by a factor of two. Thus, for an n-city tour, there are n!/(2*n) distinct circuits to consider. [19]

A Neuro-Fuzzy system is a fuzzy inference system that uses a learning algorithm derived from neural network theory to determine the parameters: fuzzy sets, membership functions (MFs) and fuzzy rules by processing the data samples. The learning capabilities of neural networks make them a prime target for a combination with fuzzy systems in order to automate or support the process of developing a fuzzy system for a given task. To enhance the knowledge acquisition, neural networks are extended to extract automatically fuzzy rules from numerical data. The heuristic learning procedure operates on the local information; it takes the semantic properties of the underlying fuzzy system into account which results in constraints on the possible modifications applicable to the system parameters. A Neuro-Fuzzy system can be always (i.e. before, during and after learning) interpreted as a system of fuzzy rules. It is also possible to create the system out of training data from the scratch, as it is possible to initialize it by prior knowledge in form of fuzzy rules. The fuzzy rules encoded within the system represent vague samples and can be viewed as prototypes of the training data. [17]

Neuro–fuzzy computing enables us to build a more powerful intelligent decision-making system by combining the advantages of an artificial neural network with the fuzzy modelling of imprecise and qualitative knowledge. [18]

Training of fuzzy system was performed by a hybrid back propagation (supervised learning) and least-mean-square algorithm, and for optimizing the number of fuzzy rules, a subtractive-clustering algorithm was adopted.

II. Literature Review

TSP has been studied well by using Elastic Adjusting method [3], Simulated Annealing [4], Ant Colony Optimization Algorithm [4,13,15], Hopfield Neural Network [5,7,16], Genetic Algorithm [5], Modified Particle Swarm Optimization algorithm [8], Expanding Self Organizing Map [8] etc. The brief study of how these techniques were applied to solve TSP is discussed below:

Author α : Pursuing M.Tech, Computer Science & Engineering, DCRUST, Murthal, Sonepat. E-mail : nitin.jain03@yahoo.in
Author Ω : Assistant Professor, Department of Computer Science & Engineering, DCRUST, Murthal, Sonepat.
E-mail : suman2222@yahoo.com

In [3] the application of multi-pattern Elastic Adjusting method, to solve the TSP, is discussed. The method is enlightened by the synapse intensifying and adjusting mechanisms in biological neural network, which dynamically modulated the encoded gene patterns in chromosomes, guided GA to coordinate exploration and exploitation, therefore it tried to probe and evaluate the optimizing trends of static problem from its own dynamic change and the results of the experiments for TSP proved that this method has good optimizing performance. In [4] XuZhihong, SongBo and GuoYanyan discussed how Simulated Annealing-Ant Colony hybrid algorithm can be applied to solve the TSP. Based on the theoretical and practical study, the performance differences between parallel algorithms and traditional algorithms were analyzed, the feasibility of proceeding parallel algorithm to solve the TSP in cluster of PCs was analysed and some meaningful conclusions about parallel efficiency were achieved.

In [5] Junyan Liu, Zhuofu Wang, Honglian Yin and Wangling Qiu discussed the application of Genetic Algorithm and Hopfield Neural Network in solving the TSP. The optimization methods of GA in artificial neural network weight, structure and algorithm were introduced and GA-HNN was established. In GA-HNN, global property of GA and the parallelisms of artificial neural networks were combined, so the method retained the global stochastically searching ability of GA and the robustness and self-learning ability of HNN. The application of this method to the optimization of TSP has the advantages of calculation stabilization, prompt convergence and preferable precision. In [6] it is discussed how modified Particle Swarm Optimization algorithm can be applied to solve the TSP. This method involved the cooperative mechanism among the individuals such as the particles also learn from other individuals according to certain probability. Additionally, the mutation of velocity has been added according to the concept of the bird flying off at a tangent in the nature. This kind of tentative study behavior helped to find the global optimum solution more frequently. The numerical simulation results justified the effectiveness and efficiency of the proposed method.

In [7] it is discussed how Continuous Hopfield Network can be applied to solve the TSP. The energy function to be minimized was determined by both the constraints for a valid solution and total length of the touring path. Since setting of parameters in energy function was crucial to the convergence and performance of the network. The role of each parameter was analyzed and criteria for choosing these parameters were described. An iterative computation algorithm of CHN was used. In [8] it is discussed how the expanding self-organizing map (ESOM) can be used to handle the Euclidean TSP. By incorporating its neighborhood preserving property and the convex-hull property of the TSP, expanding SOM was introduced. In each learning iteration, the ESOM draws the excited neurons close to the input city, and in the meantime pushes them towards the convex-hull of cities cooperatively. The ESOM may acquire the neighborhood preserving property and the convex-hull property of the TSP, and hence it can yield near-optimal solutions. Its feasibility is analyzed theoretically and empirically.

In [10] it is discussed how Neural Network Divide and Conquer strategies can be applied to solve the TSP. The concepts of Adaptive Resonance theory and Self Organizing Maps were applied. The results demonstrated that neural networks are capable of solving such an optimization problem in the quarter-million city, range with reasonable computational costs. In [11] it is discussed how an efficient genetic algorithm (GA) can be applied, to solve the TSP, with precedence constraints. The key concept of the proposed GA was topological sort. Also, a new crossover operation was developed for the proposed GA. The results of numerical experiments depicted that the proposed GA produced an optimal solution and exhibited superior performance compared to the traditional algorithms.

In [12] Marco Dorigo, Luca Maria Gambardella discussed the application of ant colony system (ACS), a distributed algorithm to solve the TSP. In the ACS, a set of cooperating agents called ants cooperate to find good solutions to TSP's. Ants cooperate using an indirect form of communication mediated by a phenomena, they deposit on the edges of the TSP graph while building the solutions. The results show that the ACS outperforms other nature-inspired algorithms such as simulated annealing and evolutionary computation, and it was concluded that ACS-3-opt, a version of the ACS, augmented with a local search procedure, to some of the best performing algorithms for symmetric and asymmetric TSP's. In [16] Taiji YAMADA, Kazuyuki AIHARA and Makoto KOTANI discussed the application of Deterministic Chaotic Neural Networks to solve the TSP. The TSP was numerically analyzed using two types of model: (i) the devil's staircase model, which has relative refractoriness and the unit step output function and (ii) the chaotic neuron model, which has relative refractoriness and the continuous output function. Last, it was shown that CNN have high ability to solve TSP.

III. Mathematical Description

Traveling salesman problem is described as: given a finite number of cities dong with the distance between each pair of them, finding the shortest close tour of visiting every city exactly once and returning to the starting point. [7] A 4-city TSP can be represented by figure 1.

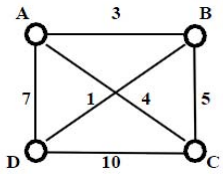


Figure 1 : 4-city TSP

The Mathematical model [7] of a N-city TSP is described as follows:

a) Distance Matrix

Given a NxN distance matrix 'D', in which D_{ij} represents the distance from i^{th} city to j^{th} city. 'D' is a symmetric matrix with all diagonal elements zeros, assuming that the traveling distance from i^{th} city to j^{th} city is equal to that from j^{th} city to i^{th} city. For figure 1, the distance matrix is shown as :

	A	B	C	D
A	0	3	4	7
B	3	0	5	1
C	4	5	0	10
D	7	1	10	0

b) Tour Matrix

Each tour can be expressed in terms of a NxN tour matrix 'V', of which x^{th} row describes the position of the x^{th} city in the current tour, the i^{th} column describes which city is chosen to visit at the i^{th} step. For figure 1, one possible trip A→C→D→B→A is shown as :

	A	B	C	D
A	1	0	0	0
B	0	0	0	1
C	0	1	0	0
D	0	0	1	0

c) Constraints on Tour Matrix

Based on the above interpretation, two constraints must be valid for tour matrix 'V' that represents a feasible or valid solution, referring to the statement of TSP.

First, each city is visited once and only once in the tour. Hence, precisely one element must equal to '1' in each row and other elements in each row must equal to '0' of the matrix 'V'. Other elements in each row must equal to '0'. i.e.

$$V_{x1} + V_{x2} + \ldots\ldots\ldots V_{xN} = 1 \text{ for } x \in \{1,2,\ldots N\}$$

Second, one city must occur in any step in the tour. So, in each column of 'V', precisely one element must equal to '1' and every other element must equal to '0'. i.e.

$$V_{1i} + V_{2i} + \ldots\ldots\ldots V_{Ni} = 1 \text{ for } i \in \{1,2,\ldots N\}$$

d) Cost Function

The total length of a tour is the sum of distance between every adjacent pair of cities. Since the element of 'V' is either 1 or 0, the total length is represented by the cost function 'C' as:

$$C' = 0.5x \sum_{i=1}^{N} \sum_{x}^{N} \sum_{(y!=x))}^{N} D_{xy} V_{x,I} (V_{y,i+1} + V_{y,i-1})$$

Additionally,

$$V_{y,N+1} = V_{y,1} \text{ and } V_{y,0} = V_{y,N}$$

The objective is to minimize the cost function under the constraints stated above.

Table 1: Number of cities, Number of possible sub-trips, Number of distinct tours

Cities n	possible sub-trips	distinct trips n!/(2*n)
4	6	3
5	10	12
6	15	60
7	21	360
8	28	2,520
9	36	20,160
10	45	1,81,440

It is shown mathematically by table 1 and graphically by figure 2, how the number of possible sub-trips and distinct tours change with the number of cities.

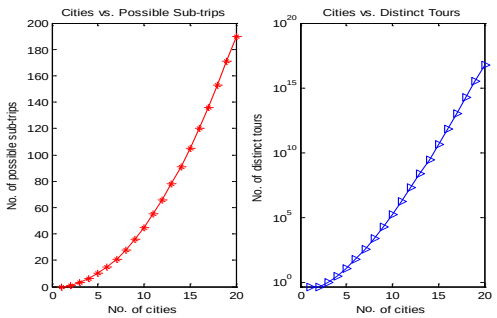


Figure 2 : Cities vs. possible sub-trips and Cities vs. distinct tours

IV. Solving TSP Using Neuro-Fuzzy System

The fuzzy system initially fuzzifies the inputs data to the values at interval [0 1] using the given set of membership functions (MFs). Next, it is inferred by the fuzzy logic through the IF–THEN rules. The basic part of the fuzzy system is the fuzzy inference engine that can be used for creating fuzzy rules. For TSP, the fuzzy rules can be represented as :

R^i : If x_1 is MF^i_1 and x2 is MF^i_2 and x3 is MF^i_3 and x4 is MF^i_4..... x_j is MF^i_j then z_i is $z_i = s^i_0 + s^i_1 x_1 + \ldots\ldots s^i_j x_j$ where R^i (i = 1, 2, . . . , L) denotes the ith fuzzy rule, x_j (j = 1, 2, . . . , n) is the jth input, z_i is the output of the ith fuzzy rule, s^i_j coefficients are the constants that are determined after training the fuzzy system, and, finally, MF^i_j is the jth fuzzy membership function of an antecedent for the ith rule, in sugeno form. [1]

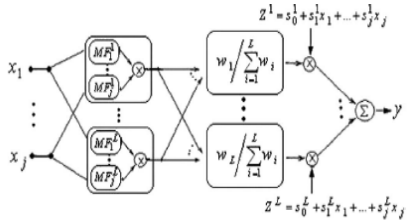


Figure 3 : Network based representation of the ANFIS structure [1]

The output of this system can be described by the following function:

$$y = [\sum_{i=1}^{L} W_i.z_i / \sum_{i=1}^{L} W_i \ ] \text{ and } W_i = \prod_{j=1}^{n} MF^i_j(x_j)$$

Here membership function used is Gaussian MF which depends on two parameters σ and c given by:

$$MF(x) = e^{-\{(x-c)2 / 2\sigma^2\}}$$

Where c is the mean and σ is the variance of the function.

The basic task of the fuzzy system is to adjust the MF parameters where the output of each fuzzy rule and the estimating number of the rules are minimized with adequate precision. The ANFIS adapts the parameters of the TSK-type inference system using a neural network. [14] In the TSK-type systems, the output is a crisp number computed by multiplying each input by a constant followed by summing up the results. To train the fuzzy system, the ANFIS employs the BP algorithm. This approach is utilized for the parameters associated with the input MFs and further uses the Least Mean Square (LMS) estimation for parameters associated with the output MFs.

Suppose that we are given an input–output pair (x, y), where x = [x1, x2, . . . , x_j]. Our goal is to minimize the cost function,

$$e = 0.5 \times \{y_{des}-y\}^2$$

Where y_{des} is the desired output. The output of each rule z_i is defined by:

$$z^i(t+1) = z^i(t) - k_z (\partial e / \partial z^i)$$

Where k_z is a learning step. To adjust the parameters of MF, namely σ and c, we start with TSK system. The derivative is given by:

$$(\partial e / \partial z^i) = (\partial e \,.\, \partial y) \times (\partial y \,.\, \partial z^i)$$

Where $(\partial y / \partial z^i) = w^i / (\sum_{i=1}^{L} w^i)$ and $(\partial e / \partial y) = y_{des}-y$

Similarly, for the jth MF of the ith fuzzy rule, the parameters c^i_j and σ^i_j are calculated.

The gradient descent algorithm based on the LMS error between the actual and desired outputs is utilized in order to adjust for the parameters. The proposed combined methods may intensify the computational complexity, but the resulting performance is noticeably improved. Furthermore, weight updating is another important step for adjusting the model parameters which is based on the BP algorithm. [1]

To increase the ability of the FIS in solving TSP, subtractive clustering [14] is utilized. This method partitions the data into clusters and generates the FIS with the appropriate number of rules required to distinguish the fuzzy qualities associated with each of the clusters. Subtractive clustering is based on a measure of the density of data points in the feature space. The idea is to find the regions in the feature space with high densities of data points. The point with the highest number of neighbors is selected as the center of a cluster. The data points within a prescribed fuzzy radius are then subtracted, and the algorithm looks for a new point with the highest number of neighbors.

When applying subtractive clustering [14] to a set of input–output data, each of the cluster centers represents a rule. To generate these rules, the cluster centers are used as the centers for the premise sets in a singleton type of rule base. We claim that our proposed method provides the appropriate number of fuzzy rules because, after each selection of a cluster center by this method, the most similar data points are placed in a cluster with specified radii. Hence, such selection approach provides the system with a relatively better choice of clusters.

V. Designing Neuro-Fuzzy Architecture

In this paper, we used the subtractive-clustering method to determine the number of fuzzy rules and the BP and LMS for the selection of the MF parameters and output of each rule. The designed network consists of several layers. The number of neurons in each layer depends on the dimension of the input vectors or the training feature set. For the system, we used four neurons in the input layer, six neurons in the hidden layer as a result of the proposed fuzzy rules, and,finally, one neuron in the output layer. Furthermore, through the implementation of subtractive clustering for each of the designed systems, the rule characteristics were determined.

As in the Sugeno type of order four, the outputs of each rule, namely z_i (for i = 1, 2,3,4, 5,6) were obtained by a combination of all mentioned inputs, where: z_i is $z_i = s^i_0 + s^i_1 x_1 + s^i_2 x_2 + s^i_3 x_3 + s^i_4 x_4$.

Where x_j (for j = 1, 2,3,4) is the jth input. The coefficients are adjusted through the LMS algorithm.

$$s^i_0 = 0 \text{ and } s^i_1 = s^i_2 = s^i_3 = s^i_4 = 1 \text{ for } (i = 1, 2, \ldots, L)$$
$$W_i = 1 \text{ for } (i = 1, 2, \ldots, L)$$

To solve a 4-city TSP (figure), there are 4 inputs, namely 'trip1', 'trip2', 'trip3', 'trip4' respectively; the complete tour will be summation of all these sub-trips. If the starting vertex is taken to be 'A', as in figure 5, there are 6 possible tours are available, hence 6 rules can be generated.

Each of the four inputs has 6 MFs, namely (A,B), (A,C), (A,D), (B,C), (B,D), (C,D) respectively. The parameters for the input MFs are set to [1.274 3], [1.274 4], [1.274 7], [1.274 5], [1.274 1], [1.274 10] respectively. The range for each input is set to [0 15] and gaussian membership function is used for each input.

Table 2 : Representation of Fuzzy Rules utilized to design Neuro-Fuzzy system

IF				THEN
x1	x2	x3	x4	z
(A,B)	(B,C)	(C,D)	(D,A)	z^1
(A,B)	(B,D)	(D,C)	(C,A)	z^2
(A,C)	(C,B)	(B,D)	(D,A)	z^3
(A,C)	(C,D)	(D,B)	(B,A)	z^4
(A,D)	(D,B)	(B,C)	(C,A)	z^5
(A,D)	(D,C)	(C,B)	(B,A)	z^6

There is only 1 output namely 'trip_distance' and 6 output MFs, namely 'output1', 'output2', 'output3', 'output4', 'output5', 'output6' respectively.

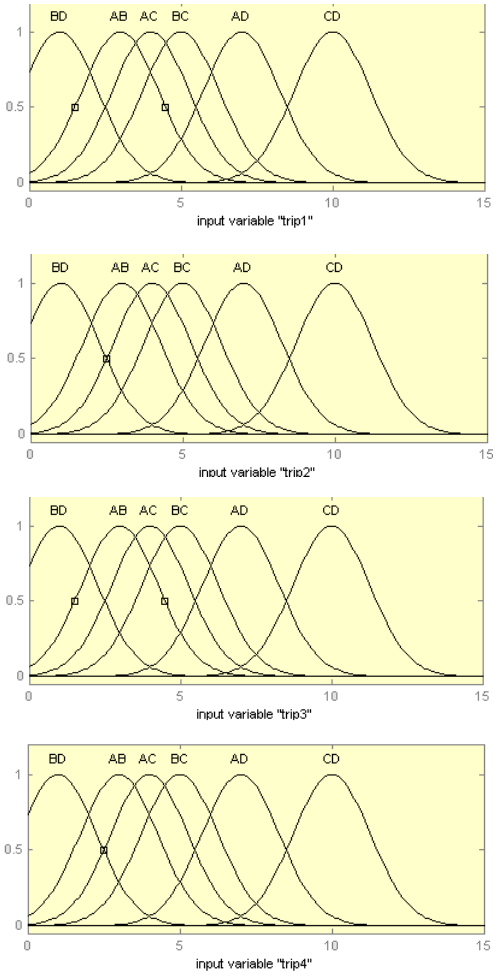


Figure 4 : Graph representing each input variable vs degree of membership

The parameters for the output MFs are set to 0.2, 0.8, 0.9, 0.8, 0.9, 0.2 respectively. The range for output is set to [0 1] and constant membership function is used for output.

The ANFIS model structure consisting of 4 inputs, 6 input MFs, 6 rules, 6 output MFs and finally 1 output is shown in figure 5.

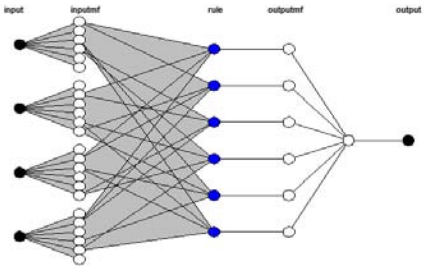
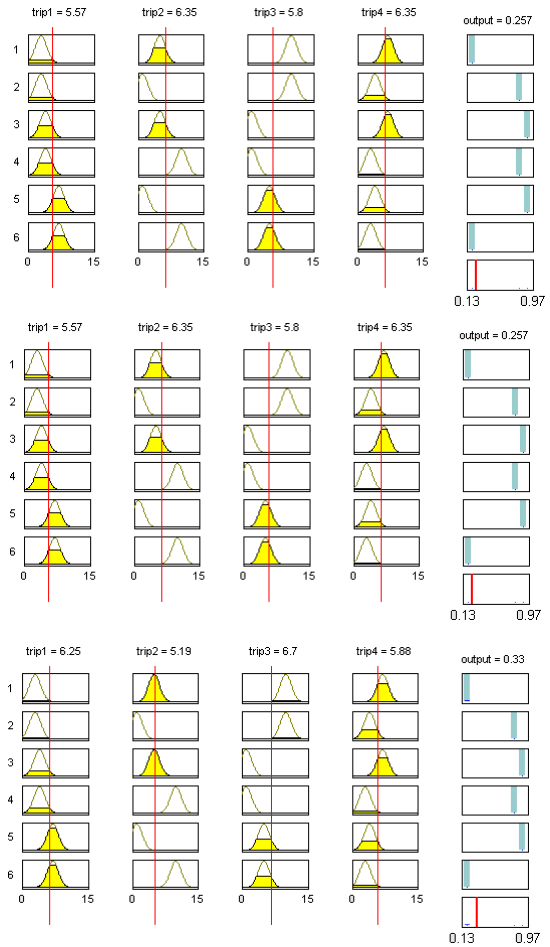


Figure 5 : ANFIS Model Structure

For the system that we designed, six fuzzy rules were obtained. Table depicts input values rules and their respective outputs. In this table, the use of MFs is represented with 6 linguistic expressions namely (A,B), (A,C), (B,C), (B,D), (C,D), (D,A). To design the system, we used "min" as the AND method and "max" as the OR method; "wtaver" as the defuzzification method.



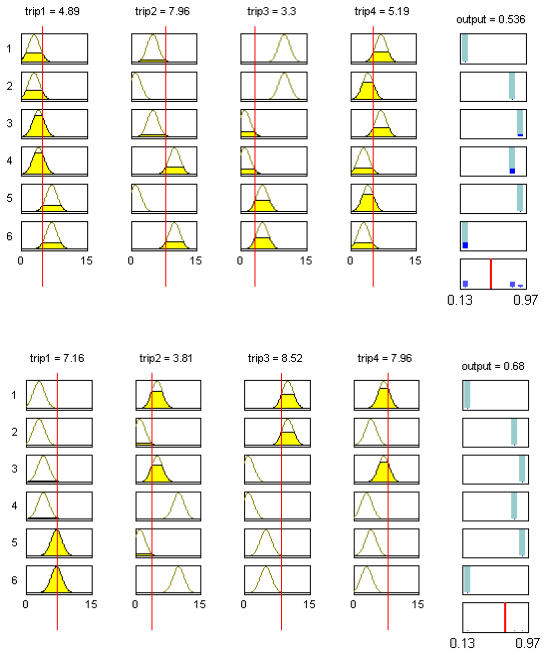


Figure 6 : Rule Viewer

The figure 6 shows that how the output values change by changing the input values, according to the rules. The respective values of the outputs obtained by changing the input values are shown in table 3. The figure 7 shows that how the respective surface diagrams can be represented.

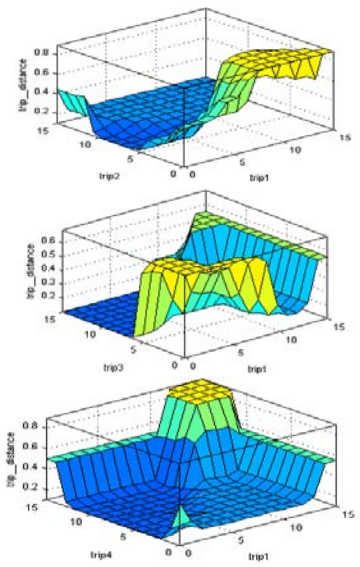


Figure 7 : Surface Viewer

VI. Experimental Results

For a 4-city TSP, figure (1,4,6); the output values are calculated by changing the values of four inputs. The output is calculated by using the concept of 'weighted average' for defuzzification.

Table 3 : Outputs vs respective input values (figure 6)

trip1	trip2	trip3	trip4	trip_distance
5.57	6.35	5.8	6.35	0.257
6.25	5.19	6.7	5.88	0.33
4.89	7.96	3.3	5.19	0.536
7.16	3.81	8.52	7.96	0.68
6.48	4.73	6.02	6.58	0.69

For testing FIS, the following matrices (4x5 & 5x6 respectively) are loaded as training data:

trainD4 =

0	3	5	4	3
3	0	6	9	6
5	6	0	7	7
4	9	7	0	4

trainD5=

0	7	36	11	9	7
7	0	13	19	23	13
36	13	0	8	42	8
11	19	8	0	10	10
9	23	42	10	0	9

The training results are shown as in table 4.

Table 4 : Training Error & Average Testing Error

Matrix	Training Error	Average Testing Error
trainD4	1.2273e-007	1.10293e-007
trainD5	1.96643e-006	2.35423e-006

ANFIS information: (4x5 training data)

Number of nodes: 47
Number of linear parameters: 20
Number of nonlinear parameters: 32
Total number of parameters: 52
Number of training data pairs: 4
Number of checking data pairs: 4
Number of fuzzy rules: 4

ANFIS information: (5x6 training data)

Number of nodes: 68
Number of linear parameters: 30
Number of nonlinear parameters: 50
Total number of parameters: 80
Number of training data pairs: 5
Number of checking data pairs: 5
Number of fuzzy rules: 5

VII. Conclusion and Future Scope

4-city Traveling Salesman Problem is solved using Adaptive Neuro-Fuzzy inference system. The Takagi-Sugeno-Kang Neuro-Fuzzy architecture is used in this application. The FIS structures were trained against a number of vague training data samples.

Finally FIS structure was trained to result in a specific output and minimizing the error. For 10 training data samples, the minimum average testing error was found to be 1.05958e-007 which is significantly small. The optimal tour was found which found the output with this error. It is shown that neuro-fuzzy approach gives more accurate results with lesser error as compared to the other techniques. Furthermore, this approach can be used in solving TSP with large number of cities.

Refrences Refrences Refrencias

1. Mahdi Khezri and Mehran Jahed, Member, 'A Neuro–Fuzzy Inference System for sEMG-Based Identification of Hand Motion Commands', © 2010 IEEE.
2. Kai Keng Ang, Cuntai Guan, Kerry Lee, Jie Qi Lee, Shoko Nioka, Britton Chance, 'Application of rough set-based neuro-fuzzy system in NIRS-based BCI for assessing numerical cognition in classroom', ©2010 IEEE.
3. Yong Song, Xianfu Chen, Yongyuan Qing, Jingchuan You, 'Elastic Adjusting Method and its application to solve static TSP', 2009 Fifth International Conference on Natural Computation, © 2009 IEEE.
4. XuZhihong, SongBo, GuoYanyan, 'Using Simulated Annealing and Ant Colony Hybrid Algorithm to Solve TSP', © 2009 IEEE.
5. Junyan Liu and Zhuofu Wang, Honglian Yin, Wangling Qiu, 'GA-Hopfield Network for Transportation Problem', © 2008 IEEE.
6. Cuiru Wang, Jiangwei Zhang, Jing Yang, Chaoju Hu, Jun Liu, 'A Modified Particle Swarm Optimization Algorithm and its Application For Solving TSP', © 2005 IEEE.
7. Helei Wu, Yirong Yang, 'Application of Continuous Hopfield network to solve the TSP8th International Conference on Control, Automation, Robotics and Vision © 2004 IEEE.
8. Kwong-Sak Leunga, Hui-Dong Jinb, Zong-Ben Xuc, 'An expanding self-organizing neural network for the traveling salesman problem', Neurocomputing 62 (2004) 267 – 292.
9. J.S.R. Jang, C.T. Sun, E. Mizutani, 'Neuro-Fuzzy and Soft Computing', A Computational Approach to Learning and Machine Intelligence' PHI, © 2003.
10. Sam Mulder, Dlonald C. Wunsch II, 'Large Scale TSP via Neural Network Divide and Conquer', © 2002 IEEE.
11. Chiung Moon, Jongsoo Kim, Gyunghyun Choi, Yoonho Seo, 'Discrete Optimization-An efficient genetic algorithm for the traveling salesman problem with precedence constraints', European Journal of Operational Research 140 (2002) 606–617.
12. Marco Dorigo, Luca Maria Gambardella, 'Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem' IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, VOL. 1, NO. 1, APRIL 1997.
13. M. Dorigo and L. M. Gambardella, 'Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem', IEEE Transactions on Evolutionary Computation, 1(1) (1997) 53–66.
14. S. Chiu, 'Method and software for extracting fuzzy classification rules by subtractive clustering,' in Proc. Biennial Conf. North Amer. Fuzzy Inf. Process. Soc., 1996, pp. 461–465.
15. L. M. Gambardella and M. Dorigo, 'Ant-Q: A Reinforcement Learning Approach to the Traveling Salesman Problem', in Proceedings of the Twelfth International Conference on Machine Learning (Morgan Kaufmann, 1995).
16. Taiji YAMADA, Kazuyuki AIHARA and Makoto KOTANI, 'Chaotic Neural Networks and The TSP', © 1993 IEEE.
17. V.K.Lamba, 'Neuro-Fuzzy Systems', University Science Press, © 2008.
18. Physica Verlag, 'Advances in Soft Computing, Introduction to Neuro-Fuzzy Systems', A Springer Verlag Company, © 2000.
19. James A. Freeman, David M. Skapura, 'Neural Networks - Algorithms, Applications, and Programming Techniques', Addison-Wesley Publishing Company, Copyright © 1991 by Addison-Wesley Publishing Company, Inc.
20. MATLAB Product Help, Matlab 7.8.0.347, R2009a, Copyright 1984-2009.

This page is intentionally left blank