Proceedings of the 9th WSEAS International Conference on Applied Mathematics, Istanbul, Turkey, May 27-29, 2006 (pp351-355)

An Efficient Growing Ring SOM and Its Application to TSP

Yanping Bai[1], Wendong Zhang[1]
[1] Key Laboratory of Instrument Science and Dynamic Measurement of Ministry of Education,
North University of China,
No 3 XueYuan Road, TaiYuan, ShanXi 030051, China

Hongping Hu[2]
[2] Dept. of Applied Mathematics,
North University of China,
No 3 XueYuan Road ,TaiYuan, ShanXi 030051, China

Abstract: - This paper presents an automatic parameters adjustment learning algorithm for self-organizing maps having growing ring topology. Like the existing SOM-like algorithm, the heuristic algorithm possesses many of these advantages of a good heuristic for the TSP solution. These advantages are easy implementation, fast computation, and production of good solutions. Computer programs developed in MATLAB for the heuristic algorithm were used to solve twelve test problems of the TSP from the TSPLIB. The experiment is showed that the results have an average of 2.4925% difference from the optimum route. We can find almost optimal route by the heuristic algorithm. The algorithm is well suited for larger instances of the TSP since it has a fast convergence and low complexity.

Key-Words: - Neural networks; Self-organizing maps; Traveling salesman problem; Combinatorial optimization; Neural computation

1 Introduction

The Traveling Salesman Problem (TSP) is one of the typical combinatorial optimization problems. It can be stated as a search for the shorted closed tour that visits each city once and only once. The decision-problem form of TSP is a NP-complete [1], thence the great interest in efficient heuristics to solve it. There are several real-life applications of the TSP such as, VLSI routing [2], hole punching [3]. And wallpaper cutting [4]. There, the research on the TSP is theoretically important.

Many of the heuristics proposed recently utilize the paradigm of neural computation or related notions [5]. However there are two types of neural network approaches for the TSP: the Hopfild-type neural networks [6] and the Kohonen-type self-organizing map (SOM-like) neural networks [7]-[12]. The underlying idea of the Hopfied-type networks is to find solutions by automatically searching for the equilibrium states of one dynamic system corresponding to the problem under consideration. The Hopfield-type networks can be successfully applied to solve small or some medium scale TSPs[15]. However, few promising solutions for general medium or large scale TSPs can be obtained. On the other, the SOM-like neural networks can handle large scale TSPs with low computation complexity. We will focus on the SOM-like neural networks in this paper.

The SOM algorithm, originally introduced by Kohonen [7], is an unsupervised learning algorithm, where the learning algorithm establish a topological relationship among input data. By simply inspecting the data values of the input cities for regularities and patterns, and then adjusting itself to fit the input data through cooperative adaptation of the synaptic weights, such a SOM brings about the localized response to the input data, and thus reflects the topological ordering of the input cities. This neighborhood preserved map then results in a tour of the TSP under consideration. From each city, the resultant tour tries to visit its nearest city. The shortest subtour can intuitively lead to a good tour for the TSP. Such a property learned by the SOM is referred to as the local optimality hereafter. Due to their low computation complexity and promising performance, the SOM-like networks have attracted a large amount of research to explore and enhance the capability of handing the TSP [9], [12]-[14]. This paper presents an efficient growing ring SOM for TSP.

The rest of paper is organized as follows: The SOM-like algorithm is briefly described in section 2. The efficient heuristic algorithm having growing ring topology is conducted in section 3. Computational experiments are presented in section 4. Conclusions are presented in Section 5.

2 The SOM-like

There are many different types of self-organizing maps; however, they all share a common characteristic, i.e. the ability to assess the input patterns presented to the networks, organize itself to learn, on its own, based on similarities among the collective set of inputs, and categorize them into groups of similar patterns. In general, self-organizing learning (unsupervised learning) involving the frequent modification of the network's synaptic weights in response to a set of input patterns. The weight modifications are carried out in accordance with a set of learning rules. After repeated applications of the patterns to the network, a configuration emerges that is of some significance.

Kohonen self-organizing map belong to a special class of neural networks denominated Competitive Neural Networks, where each neuron competes with the others to get activated. The result of this competition is the activation of only one output neuron at a given moment. The purpose of Kohonen self-organizing map is to capture the topology and probability distribution of input data [7]. This network generally involved an architecture consisting of uni or bi-dimensional array of neurons. The original uni-dimensional SOM topology is similar to a bar. A competitive training algorithm is used to train the neural network. In this training mode, not only the winning neuron is allowed to learn, but some neurons within a predefined radius from the winning neuron are also allowed to learn with a decreasing learning rate as the cardinality distance from the winning neuron increases. During the training procedure, synaptic weights of neurons are gradually changed in order to preserve the topological information of the input data when it is introduced to the neural networks.

To apply this approach to the TSP, a two-layer network, which consists of a two-dimensional input unit and m output units, is used. The evolution of the network may be geometrically imagined as stretching of a ring toward the coordinates of the cities. The input data is the coordinates of cities and the weights of nodes are the coordinates of the points on the ring. The cities are presented to the network in a random order and a competition based on Rucilidean distance will be held among nodes. The winner node is the node (J) with the minimum distance to the presenting city. J=Argmin$_j\{\|x_i$-$y_j\|_2\}$, where x_i is the coordinates of city i, y_j is the coordinates of nodes j and $\|.\|_2$ is the Euclidean distance.

Furthermore, in each iteration, the winner node and its neighbors move toward the presenting city i using the neighborhood function,

$$f(\sigma, d) = e^{(-d^2/\sigma^2)} \tag{1}$$

According to the following updated function

$$y_j^{new} = y_j^{old} + \alpha f(\sigma, d)(x_i - y_j^{old}) \tag{2}$$

Where α and σ are learning rate and neighborhood function variance, respectively. And d=min$\{\|j$-$J\|$, m-$\|j$-$J\|\}$ is the cardinal distance measured along the ring between nodes j and J, where $\|.\|$ represents absolute value and m is the number of the neurons. The neighborhood function determines the influence of a city on the neighbor nodes of the winner. We noted that the update of neighbor nodes in response to an input creates a force that preserves nodes close to each other and creates a mechanism for shortening the constructed tour.

We propose the learning rate and neighborhood function variance as follows:

$$\alpha_t = \frac{1}{\sqrt[4]{t}} \tag{3}$$

$$\sigma_t = \sigma_{t-1} \times (1 - 0.01 \times t) \tag{4}$$

Equation (3) determines the evolution of the learning rate. There is no need to define an initial value to this parameter since it depends only on the number of iterations. Equation (4) determines the evolution of the neighborhood function variance. This parameter requires an appropriate initialization, where σ_0=10 has been adopted.

3 The efficient growing ring SOM

The Efficient Growing ring SOM (EGSOM) differs from former work in this direction as no ring structure with a fixed number of a node. The structure of the EGSOM is grown based on the winning number of each node and states of its neighbors. Basic structure of the EGSOM is same as shown in Fig.2. The input layer is a 2-dimensional input space with n cities and output layer is a discrete space of nodes. Let N_j be the neighbors set of the node j where j is modulus m. The node j is characterized by an n-dimensional synaptic vector y_j and a signal counter C_j for inspection of the learning history. The EGSOM algorithm is given by the following:

Step 1 Initialization:

Let n is the number of cites and t be a discrete learning time starting from 0. The algorithm inputs

are the Cartesian coordinate of the set of cites. We give initial conditions $m(0)$, $C_j(0)$ and $y_j(0)$, where j $\{1,\ldots,m(0)\}$.

Step 2 Randomizing input signal:

Starting calculation with randomly the order of the cities after a complete cycle of iterations and label cities 1 ,…, n. Let i be the index of the city in presentation. Set i=1 and reset the inhabitation status of all nodes to false.

Inhibit [j]=false for j=1,…, $m(t)$.

Step 3 Parameters adaptation:

The learning rate α_t and the neighborhood function variance σ_t are calculated using equations (3) and (4), respectively.

Step 4 Determination of winner node:

Through a competitive procedure, select the closed node to city i based on Euclidean distance. This competition is held among only nodes, which have not been selected as a winner in this iteration. Therefore, the winner node J may be selected according to the following statement:

J=Argmin$j$||x_i-y_j|| for {j| inhibit [j]=false}

Step 5 Update of synaptic vectors and counters:

According to Equ.(1) and (2), synaptic vectors of the winner and its neighbors are updated and other synaptic vectors are preserved at time t.

$$y_j(t+1)=\begin{cases} y_j(t)+\alpha_t f(\sigma_t,d)(x_i(t)-y_j(t)) & if\ i\in N_J \\ y_j(t) & otherwise \end{cases} \quad (5)$$

Where j is modulus $m(t)$. The neighbor length of the node J is limited to 40% of $m(t)$. The signal counter of the winner is updated and other signal counters preserve their values at time t.

$$C_j(t+1)=\begin{cases} C_j(t)+1 & if\ i=J \\ C_j(t) & otherwise \end{cases} \quad (6)$$

Step 6 Insertion of a node:

Our algorithm inserts one novel cell at every T_{int} learning times hereby the map can grow. At $t=kT_{\text{int}}$, k is a positive integer, we determine one node p whose signal counter has the maximum value.

$$C_p(kT_{\text{int}})\geq C_j(kT_{\text{int}}) \quad for\ all\ j \quad (7)$$

If there exist plural maximum counter value we randomly select one of the nodes. This is an inspection of the learning history. We take a neighbor f=p-1, where f is modulus $m(t)$. A novel node r is inserted between p and f. The synaptic vector of r is initialized as follow:

$$y_r=0.5(y_p+y_f) \quad (8)$$

Counter values of p and r are re-assigned as follows:

$$C_p(t+1)=0.5C_p(t),\quad C_r(t+1)=0.5C_p(t) \quad (9)$$

The insertion interval T_{int} is a control parameter of this algorithm. After the insertion, the number of nodes increases: let $m(t+1)=m(t)+1$.

Step 7 Termination:

Let $t=t+1$. If $m(t)<1.5n$ then go to step 2, otherwise the learning is terminated.

4 Application to TSP

We apply the EGSOM algorithm to some TSP instances at http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/. At t=0, we give initial conditions:

$m(0)=50$, $C_j(0)=0$, $T_{\text{int}}=50$.

Location of each city corresponds to an input and is applied randomly to the EGSOM. As learning time goes the EGSOM grows and forms a tour route. Then we search the closest nodes from each city. Every city can obtain the distinct closed nodes hereby the tour route can be determined because the EGSOM has ring topology. After using the EGSOM algorithm to solve 12 test problems of the TSP from the TSPLIB, an experiment is showed in table 1 that the best results have an average of 2.9910% difference from the optimum route by 20 simulation runs. We can find almost optimal route by the EGSOM. Because the number of nodes $m(t)$ need only grow to 1.5n, the algorithm processing time is fast.

Table 1. The best results and percent differences from the optimum route by using the EGSOM algorithm solve 12 test problems of the TSP from the TSPLIB by 20 simulation runs

Instances	No. of cities	The optimum route.	The best route from the EGSOM	Percent difference from optimal tour	Average processing time (secs.)
eil51	51	426	431.9569	1.3983	6.2672
st70	70	675	683.1030	1.2004	9.8048
rd100	100	7910	8024.3	1.4453	15.846
lin105	105	14379	14408	0.1986	16.361
pr107	107	44303	44750	1.0092	20.913
Bier127	127	118282	119610	1.1201	26.7600
pr136	136	96772	98963	2.2640	30.936
Pr152	152	73682	74228	0.7412	39.914
kroA200	200	29368	29947	1.9715	60.494
pcb442	442	50778	55263	8.8333	285.40
Att532	532	87550	91063	4.0125	604.07
U1060	1060	224094	236900	5.7154	1689.3
Average percent difference				2.4925	233.8388

Figure 1 to 4 show the partial results obtained by the application of the EGSOM algorithm.

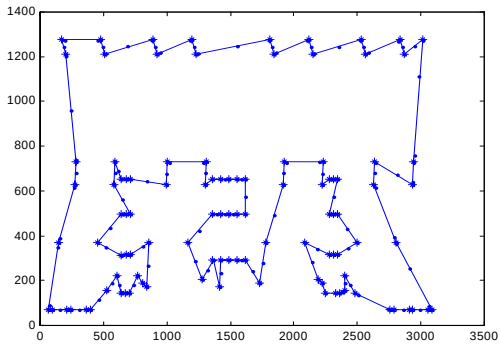

Fig. 1 EGSOM for lin105

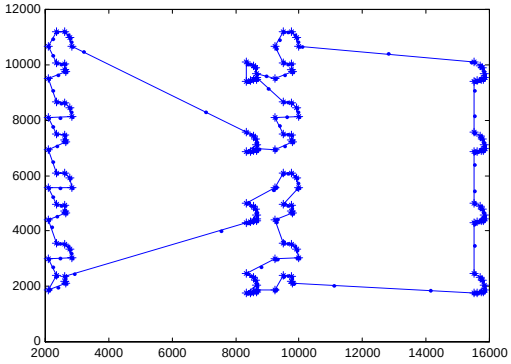

Fig. 2 EGSOM for pr152

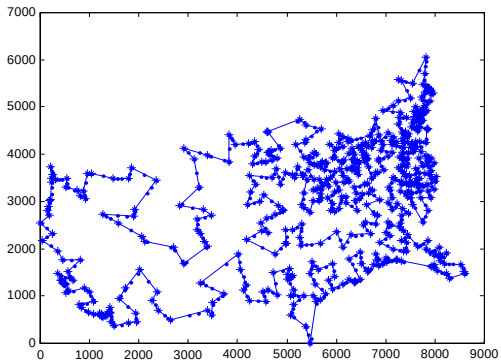

Fig. 3 EGSOM for att532

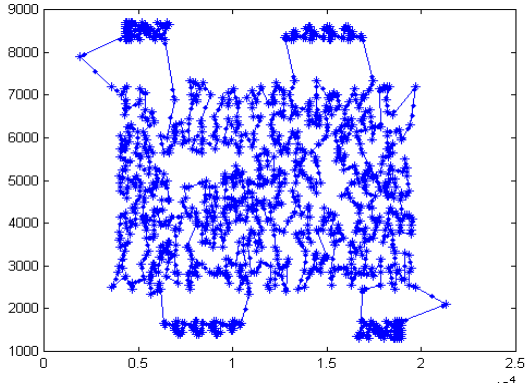

Fig. 4 EGSOM for u1060

5 Conclusions

We have presented an automatic parameters adjustment algorithm EGSOM for growing ring SOM. Computer programs developed in MATLAB for the heuristic algorithm EGSOM were used to solve twelve test problems using a standing desktop computer. Like the existing heuristic, the modified heuristic possesses many of these advantages of a good heuristic for the TSP solution. These advantages are easy implementation, fast computation, and production of good solutions. After using the EGSOM algorithm to solve twelve test problems of the TSP from the TSPLIB, an experiment is showed in table 1 that the best results have an average of 2.4925% difference from the optimum route by 20 simulation runs. We can find almost optimal route by the EGSOM. Because the number of nodes $m(t)$ need only grow to $1.5n$, the algorithm processing time is fast. Therefore, the EGSOM is well suited for larger instances of the TSP since it has a fast convergence and low complexity.

6 Acknowledgment

The authors are thankful that the research is supported by Shanxi Province's (20051006) and national (10471040) nature Science Foundations in China.

References:

[1] Papadimitriou, C. H., The Euclidean traveling salesman problem is NP-complete. *Theoretical Computer Science*, Vol 4, 1978, pp. 237-244.

[2] K. A. Smith, Neural networks for combinatorial optimization: A review of more than a decade of research, *INFORMS J. Comput.*, Vol. 11, No. 1, 1999, pp. 15-34.

[3] G. Reinelt, TSPLAB—A traveling salesman problem library, *ORSA J. Computing*, Vol.3, No. 4, 1991, pp. 376-384.

[4] G. Laporte, The vehicle routing problem: An overview of exact and approximate algorithms, *Eur. J. Oper. Res.*, Vol. 59, 1992, pp. 345-358.

[5] C. Peterson, Parallel distributed approaches to combinatorial optimization: benchmark studies on traveling salesman problem , *Neural computation*, Vol. 2, 1990, pp. 261-269.

[6] Hopfield, J. J. and Tank, D. W., Neural computation of decisions in optimization problems, *Biological Cybernetics*, Vol. 52, 1985, pp. 141-152.

[7] Kohonen, T., Self-organized formation of topologically correct feature maps, *Biol. Cybern.*, Vol. 43, No. 2, 1982, pp. 59-69.

[8] N. Aras. B.J. Oommen. I.K. Altinel., Kohonen Network incorporating explicit statistics and its application to the traveling salesman problem, *Neural Networks,* Vol. 12, 1999, pp. 1273-1284 .

[9] Somhom, S., Modares, A, & Enkawa, T., competition-based neural network for the multiple traveling salesman problem with minmax objective. *Computers and Operations Research*, Vol. 26, 1999, pp. 395-407.

[10] Favata, S. & Walker, R., A study of the application of Kohonen–type neural networks to the traveling salesman problem. *Biological Cybernetics*, Vol. 64, 1991, pp. 463-468.

[11] Hueter, G.J., Solution of the traveling salesman problem with an adaptive ring. *Proceeding of the IEEE International Conference on Neural Networks*, 1988 , No. I, pp. 85-92.

[12] Fort, J. C., Solving combinatorial problem via self-organizing process: an application of the Kohonen algorithm to the traveling sales-man problem. *Biological Cybernetics,* Vol. 59, 1988, pp. 33-40 .

[13] Ange'niol, B., Vaubois, C. and Le Texier, J.Y., self-organizing feature maps and the traveling salesman problem. *Neural Networks*, Vol. 1, 1988, pp. 289-293.

[14] H. D. Jin K. S. Leung, M. L. Wong, Z. B. Xu, An efficient self-organizing map designed by genetic algorithms for the traveling salesman problem, *IEEE Trans. On systems, man, and cyber. P. B, Cyber.*, Vol. 33, 1997, pp. 877-888.

[15] S. Abe, Convergence acceleration of the Hopfild neural network by optimization integration step sizes, *IEEE Trans. Syst., Man, Cybern. B,* Vol. 26, pp. 1996, pp. 194-201.