SYNOPTIC STUDY OF
KOHONEN’S APPROACH FOR THE
APPROXIMATE SOLUTION OF TSP

INTRODUCTION

A detailed approach for solving TSP using Hopfield network has been described in previous report. Here an overview is given for the possible second approach for the solution of TSP, i.e. using Kohonen Self Organized Map. As SOM name suggests converges to the required result on its own from the initial input state to the desired output, this approach can be very easy.

KOHONEN NETWORK

Kohonen’s self organizing maps can be used for the traveling salesman’s problem. Here each city considered for tour is referenced by its x and y coordinates. To each city there corresponds a neuron. The neurons are placed in a single array, unlike the two-dimensional array used in the Hopfield approach. The first and the last neurons in the array are considered neighbor.

There is a weight vector for each neuron, and it also has two components. The weight vector of a neuron is the image of the neuron in the map, which is sought to self organize. There are as many input vectors as there are cities, and the coordinate pair of a city constitutes an input vector. A neuron with a weight vector closest to the input vector is selected. The weights of neurons in a neighborhood of the selected neuron are modified, others are not. A gradually reducing scale factor is also used for the modification of weights.

There are as many neurons in the network as the number of cities. They have random initial conditions. Thus their initial coordinates are randomly generated. In each iteration the neuron closest to the corresponding city is found and the weights are updated.

That a tour of shortest distance results from this network operation is apparent from the fact that the closest neurons are selected. It is reported that experimental results are very promising. The computation time is small and solution is near optimal.

ALGORITHM FOR
KOHONEN’S APPROACH

A gain parameter of $\lambda$ and a scale factor $q$ are used while modifying the weights. A value between 0.02 and 0.2 is selected. A distance of a neuron from the selected neuron is defined to be an integer between 0 to n-1, where n is the number of cities for the tour. This means that these distances are not necessarily
actual distance between the cities. They could be made representative of the actual distances between the cities. They could be made representative of the actual distance by using a Squashing function similar to the Gaussian density function. The distance is denoted by $d_j$ for neuron $j$.

The steps for the algorithm are

- The initial weights can be set as

$$ r[i,j] = e^{(-\text{dist}(i, j)^2)/(2 \theta)} $$

Theta is non-negative value.

- Find the weight vector for which the distance from the input vector is smallest.

- Modify weights using

$$ W_{j_{\text{new}}} = W_{j_{\text{old}}} + (i_{\text{new}} - W_{j_{\text{old}}}) g(\lambda, d_j) $$

Here,

$$ g(\lambda, d_j) = \exp(-d_j^2/\lambda)/\sqrt{2}. $$

- Reset $\lambda$ as $\lambda(1-q)$.

- The Gaussian density function that can be used as squashing function is

$$ F(d, \lambda) = \exp(-d^2/\lambda)/\sqrt{2}. $$

**CONCLUSION**

- Though I have not implemented the algorithm, the results seems promising considering the proofs of the equations given. Kohonen approach surely will give a good result for TSP though this is not as well suited for TSP as Hopfield network.

- The input here is $x$ and $y$ coordinate of city instead of distance between cities as in Hopefiled.

- The results surely will be obtained much faster than simple algorithm.

- It is still to be seen how generalized this algorithm is and can it be modified easily as Hopfield network for other NP-complete problems

**REFERENCES**

- Timothy Masters, Practical Neural Network Recepies in C++.
- Papers by Eric Davalo and Patrick Naim.