A Self-Organising Neural Network for the Travelling Salesman Problem that is Competitive with Simulated Annealing

Marco Budinich
Center for Neural Networks - King's College London
(Permanent address: Dipartimento di Fisica & INFN, Via Valerio 2, 34127 Trieste, Italy)
E-mail: mbh@trieste.infn.it

(Appeared in: Neural Computation Volume 8, Issue 2 - February 15, 1996)

Abstract - Unsupervised learning applied to an unstructured neural network can give approximate solutions to the travelling salesman problem. For 50 cities in the plane this algorithm performs like the elastic net of Durbin and Willshaw [Durbin 1987] and it improves when increasing the number of cities to get better than simulated annealing for problems with more than 500 cities. In all the tests this algorithm requires a fraction of the time taken by simulated annealing.

1 - Introduction

The Travelling Salesman Problem (TSP) is the archetype of NP-complete problems and has attracted considerable research effort (see e. g. [Lawler 1985]). In one of its simplest form there are n cities in the plane and the problem is to find the shortest closed tour that visit each city once. While all exact solutions take a time that grows exponentially with n there are many heuristic algorithms for practical instances [Johnson 1990].

In this paper I propose a neural network algorithm that produces approximate solutions for the TSP. Unlike more traditional neural network approaches [Hopfield 1985] here the solution is given by the self-organising properties of unsupervised learning.

The basic idea comes from the observation that in one dimension the exact solution to the TSP is trivial: always step to the nearest unvisited city. Consequently, given a TSP with cities in the plane, if one can map them "smartly" to a set of cities distributed on a circle, he will easily find the shortest tour for these "image cities" and a visit order for them: this visit order will give also a tour for the original cities. Obviously if the map preserves perfectly all the distance relations of the cities this

will be the shortest TSP tour i. e. the exact solution. Unfortunately a perfectly neighborhood preserving map does not exist in general but it is reasonable to conjecture that the better the distance relations are preserved the better will be the approximate solution found. In this way the original TSP is transposed to the search of a good neighborhood preserving map.

There are several ways to search for such a map: Durbin and Willshaw [Durbin 1987] did it relaxing a fictitious physical system, an “elastic net”. Another method is to use a self-organising neural net to find neighborhood preserving maps. These nets, proposed to model the self-organising feature maps in the brain [Von der Malsburg 1973, Kohonen 1984], can find good neighborhood preserving maps through unsupervised learning. In this way the TSP is ultimately solved by unsupervised learning.

In conclusion there are two steps to get a solution for the TSP: the first is to teach the problem to a self-organising neural net that, while learning, builds up the neighborhood preserving map. In the second step, from the solution for the image cities, one obtains a tour for the original TSP.

In what follows I will present in detail this algorithm and compare it with the well established simulated annealing technique [Kirkpatrick 1983], and, given the similarity of the approaches, with the elastic net [Durbin 1987].

2 - Solving the TSP with self-organizing maps

The TSP considered here is given by n cities randomly distributed in the (0,1) square. The network has two input neurons that pass the $(x, y) = \bar{\zeta}$ coordinates of the cities to n output neurons. The output neurons form a ring; the distance D(i, j) between neurons i and j is one plus the minimum number of neurons between i and j. Each output neuron has just two weights: $(w_x, w_y) = \bar{w}$ and, in response to the input (x, y) from city $\bar{\zeta}$, gives an output $o = \bar{\zeta} \cdot \bar{w}$. Figure 1 gives a schematic view of the net.

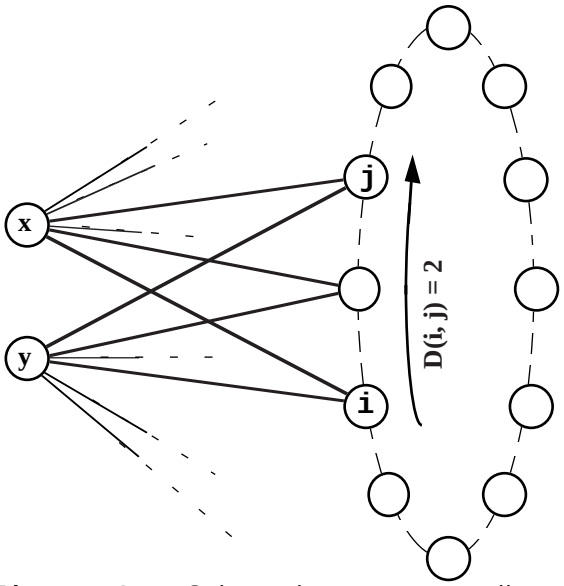

Figure 1 Schematic net: not all connections from input neurons are drawn. In this example D(i, j) = 2.

For the neural network the square represents the input space and the cities are the patterns to be learned. After learning, the network maps the two dimensional input space onto the one dimensional space given by the ring of neurons.

The standard learning algorithm for this networks is [Kohonen 1984]:

1 set the weights to initial random values in [0,1];

2 select a city at random, say $\vec{\zeta}$, and feed its coordinates to the input neurons;

3 find the output neuron with maximal output, say m[1];

4 train m and its neighbors up to a distance d with Hebb rule; the training affects 2 d + 1 neurons:

$$\vec{w}_i' = \vec{w}_i + \alpha (\vec{\zeta} - \vec{w}_i) \qquad \forall\ i :\ D(i, m) \le d$$

5 update the parameters d and α according to a pre-defined schedule and, if the learning loops are not yet finished, go to 2.

A preliminary study of the learning parameters gave this recipe to obtain the best performances:

- the number of output neurons is fixed and equal to the number of cities n;
- 50 learning loops for each city of the problem;
- the learning constant α starts from 0.8 and its value is decreased linearly at each learning loop reaching zero at the last iteration;
- the most active neuron m is trained with learning constant α, its neighbors are trained with a value of α that decreases linearly with D(i, m) and vanishes for neurons at distance greater than d from m;
- the distance of update d is 6.2 + 0.037 n at the start and is decreased linearly to 1 in the first 65% of the iterations. For the successive 35% of the learning loops d remains constant at 1.

After learning, the net maps neighboring cities to neighboring neurons: for each city its image is given by the neuron with maximal activity. One easily finds the shortest tour for the images of the cities that, in turn, gives a tour for the TSP.

This straightforward version of the learning algorithm has a major flaw. The map it produces is not injective: many cities can be mapped to the same neuron (this happens for a fraction between 45% and 50% of the total number of cities n for $10 \le n \le 1000$). When two or more cities are mapped to the same neuron one cannot say which of them has to come first in the tour and this problem substantially reduces the performances of this algorithm [Angéniol 1988, Favata 1991].

---

[1]This definition is ambiguous unless city and weight vectors are somehow normalised. Since both cities and weights define points in the plane, the problem can be circumvented by defining the most active neuron for city $\vec{\zeta}$ as the neuron which weights define the nearest point to $\vec{\zeta}$. Simple algebra shows that the two definitions are equivalent.

3 - A new TSP algorithm

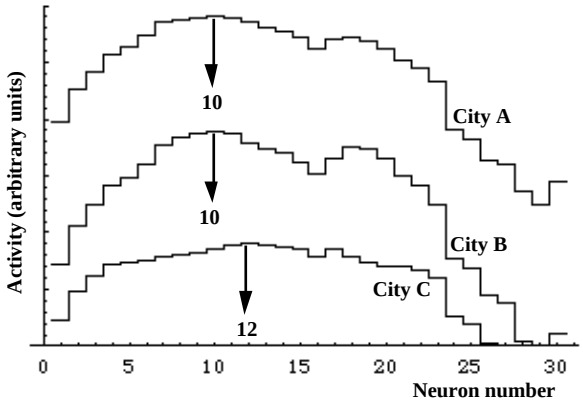

Figure 2 3 activity profiles for a net with 30 neurons obtained presenting three different cities (of the 30 of the original TSP). Cities A and B both give maximal activity in neuron number 10.

A decisive insight comes from the "activity profile" of the net: a plot of the neuron outputs in response to a given city. Figure 2 contains the activity profiles for 3 cities on a net with 30 neurons. Cities A and B both produce the maximal activity on the 10^{th} neuron.

When mapping cities to neurons we associate to each city a coordinate along the ring. The standard choice of the maximal acting neuron produces an integer coordinate that is the modal value of the activity profile.

A better choice turns out to be an averaging process on the neurons to obtain a real coordinate: for example a weighted coordinate calculated with the maximal active neuron and its nearest neighbors using their activities as weights. In this way each city is mapped to a point with a real coordinate along the ring, the map becomes injective and the ambiguities in the tour disappear.

This new mapping produces substantially shorter tours assuring that there is valuable information also in the neurons near to the most active one. The average not only uses this additional information but is also a better and more plausible estimator of the city image than the modal value since it is a linear function of the neuron activities and it makes easier to imagine a further layer of neurons doing this job.

In what follows I try to assess the effectiveness of this algorithm comparing it with two other stochastic ones: simulated annealing [Kirkpatrick 1983] and the elastic net algorithm [Durbin 1987].

4 - Performances

It is known that comparing the performance of stochastic TSP algorithms is not indisputable especially when the length of the shortest tour is not available [Johnson 1990]. I took a rather conservative approach comparing the average length obtained in 10 runs of each algorithm on the same problem keeping separately track of the CPU

time used. This solution is less subject to fluctuations than, for example, a comparison based on the minimal length obtained by an algorithm in a fixed amount of CPU time.

The implementation of simulated annealing used in these tests is that published in [Müller 1990] which is tailored to the TSP since it uses exchange terms inspired by Lin and Kernigham heuristics [Lin 1973]. With the chosen annealing factor of 0.95 the algorithm gave better performances than those quoted by Durbin and Willshaw [Durbin 1987] on the same problems. Figure 3 is a test of its stability and the lower, dashed, curve shows that its performances remain reasonably constant, with respect to the theoretical lower bound of $0.749\sqrt{n}$, when the number of cities n varies between 10 and 1000.

Johnson [Johnson 1990] performed extended comparisons on the performances of various TSP heuristics including simulated annealing (even if not in the implementation used in this work). He showed that, when the performances are measured with the average of different runs, simulated annealing is favored and, even if in longer times, beats the most reputed heuristics like that of Lin and Kernigham [Lin 1973] and, *a fortiori*, 2-opt and 3-opt.

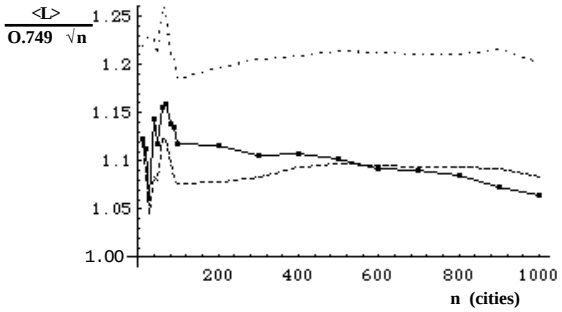

Figure 3 Ratio of the average tour length obtained in 10 runs over the theoretical lower bound of $0.749\sqrt{n}$ (thick curve) . The upper, dotted, curve shows the results obtained from (my implementation of) Favata and Walker algorithm [Favata 1991], while the lower, dashed, curve shows results from simulated annealing.

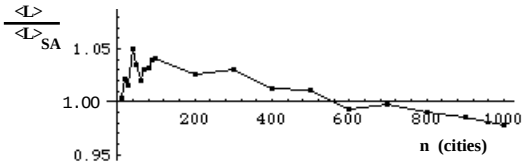

Figure 4 Ratio of the average tour length obtained in 10 runs from this algorithm over those given by simulated annealing; the number of cities varies between 10 and 1000.

Figure 4 contains the ratio of the average tour lengths of this algorithm over those obtained from simulated annealing. The neural net produces shorter tours than those of simulated annealing for problems with more than $n = 500$ cities.

It is interesting to speculate on the properties of this algorithm as depending on n, the number of cities of the problem. The amount of memory grows like $\mathbf{O}(n)$ while the time employed grows at most like $\mathbf{O}(n^2)$ since both the learning loops and the search of the most active neuron depend linearly on n. However this is probably just an upper bound since figure 3 shows that this prescription for the number of iteration gives

performances that increase with n. Consequently it is possible that, to maintain constant performances, one can use a number of learning loops that increases less than linearly with n.

Comparing the CPU time of the runs used for figure 3 it appears that, for 100 cities, this algorithm is 10 time faster than simulated annealing and it is 3 times faster for 1000 cities producing, in this case, better average performances. This decrease of the ratio is probably due to the non uniform performances when varying the number of cities. It is remarkable that this net can perform better than simulated annealing in its best achievement (average length) compensating at the same time its main weakness: slowness.

A different test of the performances was made on the 5 city sets used by Durbin and Willshaw [Durbin 1987]. To test the algorithm in conditions as similar as possible to those of the elastic net the parameters of the net were tuned for 50 cities, and the average performances were taken over 5 runs. Table 1 contains a comparison of the performances and shows that they are substantially equal but the CPU times are rather different. Since exact time data were unavailable for comparisons I considered the time used by each algorithm relatively to the time taken by simulated annealing. Whereas Durbin and Willshaw report a time 30% longer than that employed by simulated annealing this net is 10 times faster than simulated annealing.

City set	[Durbin 1987]	This algorithm	Difference (%)
1	5.98	5.975	- 0.08 %
2	6.03	6.110	+ 1.33 %
3	5.74 [5.70]	5.737	- 0.05 % [+0.65]
4	5.90 [5.86]	5.830	- 1.19 % [-0.51]
5	6.49	6.583	+ 1.43 %

Table 1 Comparison of average tour length in 5 runs for the same problems.

5 - Conclusions

Even if this is the first time a neural network algorithm proves to be competitive with simulated annealing, deeper analysis is needed to establish if it has any relevance as a serious TSP heuristic. Just to mention another limitation it is well known that the TSP's with random cities in the plane are relatively easy.

Nevertheless the results show that this net works satisfactorily on a provably difficult problem and especially in large sizes: a not so frequent quality in neural networks. Even more important is that the results derive from a well established learning procedure applied to a previously unstructured net. This poses this approach in a favorable position when compared to those that solve the TSP relaxing a finely pre-tuned neural network [Hopfield 1985].

It is also intriguing that the theory behind this algorithm, intimately connected to the self organising processes [Erwin 1992], is today not really understood (even if the ordering theorem of Kohonen [Kohonen 1984] can be extended to the d to 1 dimensional case [Budinich 1995]).

References:

[Angéniol 1988] Angéniol B., de La Croix Vaubois G. and Le Texier J.-Y., *Self Organising Feature Maps and the Travelling Salesman Problem*, Neural Networks **1** (1988) pp. 289-293;

[Budinich 1995] M. Budinich and J. G. Taylor: *On the Ordering Conditions for Self-Organising Maps*, Neural Computation **7** #2 (March 1995), pp. 284-289;

[Durbin 1987] Durbin R. and Willshaw D., *An Analogue Approach to the Travelling Salesman Problem using an Elastic Net Method*, Nature **336** (1987) pp. 689-691;

[Erwin 1992] Erwin E., Obermayer K. and Schulten K., *Self Organising Maps: Ordering, Convergence Properties and Energy Functions*, Biological Cybernetics **67** (1992) pp. 47-55;

[Favata 1991] Favata F. and Walker R., *A Study of the Application of Kohonen-type Neural Networks to the Travelling Salesman Problem*, Biological Cybernetics **64** (1991) pp. 463-468;

[Johnson 1990] Johnson D., in: *Proceedings of the 17th Colloquium on Automata, Languages and Programming*, (1990) Springer-Verlag New York, pp. 446-461;

[Hopfield 1985] The first algorithm of this kind was introduced by: Hopfield J.J. and Tank D.W., *"Neural" Computation of Decisions in Optimization Problems*, Biological Cybernetics **52** (1985) pp. 141-152;

[Kirkpatrick 1983] Kirkpatrick S., Gelatt C.D. Jr and Vecchi M.P., *Optimization by Simulated Annealing*, Science **220** (13 May 1983) pp. 671-680;

[Kohonen 1984] Kohonen T., *Self-Organisation and Associative Memory*, 1984 (3rd Ed. 1989) Springer-Verlag Berlin Heidelberg;

[Lawler 1985] Lawler E.L., Lenstra J.K., Rinnoy Kan A.G. and Shmoys D.B. (editors), *The Traveling Salesman Problem - A Guided Tour of Combinatorial Optimization*, John Wiley & Sons, New York 1990, IV Reprint, pp. x 474;

[Lin 1973] Lin S. and Kernigham B.W., Operations Research **21** (1973) pp. 498-516;

[Müller 1990] Müller B. and Reinhardt J., *Neural Networks*, 1990 Springer-Verlag Berlin Heidelberg;

[Von der Malsburg 1973] Von der Malsburg Ch., *Self-Organising of Orientation Sensitive Cells in Striate Cortex*, Kybernetik **14** (1973) pp. 85-100.

ACKNOWLEDGEMENTS: I want to acknowledge the many, extremely fruitful, discussions with John G. Taylor at King's College. I also thank warmly King's College, the British Council and the "Consiglio Nazionale delle Ricerche" for supporting my visit at King's and D. Willshaw and M. Simmen for kind hospitality and for providing the TSP's used in [Durbin 1987].