Control and Cybernetics

vol. **31** (2002) No. 2

Neural network models for combinatorial optimization: A survey of deterministic, stochastic and chaotic approaches

by

K. A. Smith[1], J.-Y. Potvin[2], and T. Kwok[1]

[1] School of Business Systems, Faculty of Information Technology, P.O. Box 63B, Monash University, Victoria 3800, Australia

[2] Département d'Informatique et de Recherche Opérationnelle Universite de Montreal, C.P. 6128, Succ. Centre-Ville, Montreal (Quebec) H3C 3J7, Canada

Abstract: This paper serves as a tutorial on the use of neural networks for solving combinatorial optimization problems. It reviews the two main classes of neural network models: the gradient-based neural networks such as the Hopfield network, and the deformable template approaches such as the elastic net method and self-organizing maps. In each class, the original model is presented, its limitations discussed, and subsequent developments and extensions are reviewed. Particular emphasis is placed on stochastic and chaotic variations on the neural network models designed to improve the optimization performance. Finally, the performance of these neural network models is compared and discussed relative to other heuristic approaches.

Keywords: combinatorial optimization, neural networks, Hopfield networks, self-organizing maps, deformable templates, chaos.

1. Introduction

Combinatorial optimization problems (COPs) involve searching for the best possible combination of binary valued-variables, in order to simultaneously minimize a cost function and satisfy a set of constraints. COPs arise naturally in the process of mathematical modeling of many practical problems from industry, such as transportation routing, production line scheduling, and frequency assignment in mobile communications. The need to find rapid solutions to these problems, particularly in a changing environment, is important to the efficiency of many industries. The benefits of even a small improvement in the quality of the solutions are also significant. Thus both the quality and speed of solution to practical COPs are critical issues affecting the usefulness of the solution. Due

to the combinatorial complexity of these problems however, an exact solution (provably optimal) is unlikely to be found within a useful period of computation time using traditional mathematical programming techniques. Instead, heuristic approaches are preferred that aim to provide a near-optimal solution sufficiently rapidly for it to be useful.

For over a decade, researchers have attempted to solve COPs using neural networks as an alternative heuristic approach to techniques such as simulated annealing, tabu search, and greedy search. Neural networks are simple mathematical models, inspired by the interactions between neurons in the human brain, that have been applied successfully to a variety of problems including prediction, classification, and pattern recognition. Neural computing was originally developed to provide a fundamentally new and different approach to information processing, when an algorithmic procedure for solving a problem is not known. As opposed to programmed computing, neural networks are capable of internally developing information processing capabilities for solving a problem when fed with appropriate information about the problem. Thus, they are often referred to as learning or adaptive models.

The history of neural computing dates back to the paper of McCulloch and Pitts (1943), when simple types of neural networks were shown to be able to learn arithmetic or logical functions. Important successes were witnessed in the late 50's and early 60's, with the development of the Perceptron model and the first neuro-computers (Rosenblatt, 1958). By the end of the 60's, however, the field collapsed: a book by Minsky and Papert (1969), demonstrating that even the simple exclusive-or logical function could not be implemented with a Perceptron, was devastating and diverted away research funding. After a dark period, neural networks emerged again in the early 80's with the support of John Hopfield, a renowned scientist, and the publication of an important book by Rumelhart and McClelland (1986), which introduced the backpropagation neural network model to the scientific community. This model extended the capabilities of its ancestor, the Perceptron, allowing it to learn a much larger class of functions (including the exclusive-or logical function). Since that time, the field has continually expanded.

The first successes with neural networks were reported for the most part in pattern recognition, classification and prediction tasks. Application of neural networks to COPs however dates back to 1985 when Hopfield and Tank solved small instances of the traveling salesman problem (TSP[1]) with a Hopfield neural network (Hopfield and Tank, 1985). Because of the simplicity of its formulation, the TSP has always been a fertile ground for new solution ideas. Consequently, it is not surprising that many problem-solving approaches inspired by neural

[1] The TSP is a classical combinatorial optimization problem, which is simple to state but difficult to solve. Basically, the objective is to find the shortest possible tour (or Hamiltonian cycle) through a set of N vertices so that each vertex is visited exactly once. This problem is known to be NP-hard (Garey and Johnson, 1979; Nemhauser and Wolsey, 1988), and cannot be solved exactly in polynomial time.

networks have been applied to the TSP (Potvin, 1993). The other main class of neural network, known as deformable templates and including models such as the elastic net and self-organizing map, has also focused on solving the TSP. Deformable template approaches exploit the geometry of the TSP, and tend to be more efficient at solving larger TSPs than the Hopfield neural network. Their generalization to non-geometric problems, however, is quite limited.

Since the early neural models much research has been conducted and many new models for solving COPs have emerged over the last decade or so (Smith, 1999). These can be broadly categorized as Gradient-Based Neural Networks (including the Hopfield neural network), and Deformable Template Neural Networks (including the elastic net and self-organizing maps). Within each of these categories, there are many extensions and variations that have been proposed to improve the optimization capabilities of the models, using deterministic, stochastic or chaotic mechanisms. The developed models have been evaluated on a wide range of problems in addition to the TSP, such as problems in routing and transportation, scheduling, cutting stock and packing, timetabling, telecommunications, and many others (Smith, 1999; Burke and Ignizio, 1992). In some cases, the results obtained are competitive with those reported for alternative techniques. In other cases, the results are not yet convincing. It is clear that the computational paradigm of neural networks, which is inherently parallel, distributed and adaptive cannot be fully exploited on current computer hardware. Their behavior must be simulated, thus leading to excessively large computation times. The development of suitable hardware for these models (often called neuro-computers) would thus be an important step toward their full recognition.

The purpose of this paper is to provide a tutorial on the use of neural networks in combinatorial optimization. The two main approaches, Gradient-Based Neural Networks and Deformable Template Neural Networks are reviewed by presenting the basic models and some of their variations. These variations include stochastic and chaotic models. A comparison between the neural models and against more traditional heuristic approaches is also presented.

2. Gradient-based neural networks

In his seminal paper of 1982, John Hopfield described a new way of modeling a system of neurons capable of performing computational tasks (Hopfield, 1982). Using a collection of binary-state neurons and a stochastic updating algorithm, these computational tasks were initially related to storage and retrieval of embedded memories. The computational capabilities of Hopfield's original model were expanded in Hopfield (1984) when he proposed a continuous version, and proved convergence of the model by demonstrating that the dynamics of the model minimized a constructed Liapunov function over time. From here, it became clear that Hopfield networks could be used to minimize any function provided the network parameters were set appropriately. The fact that the con-

tinuous version of the Hopfield network was designed to be implemented using electrical circuits also promised rapid computational ability.

This section first presents the two Hopfield neural network models: the discrete and stochastic model of 1982, and the continuous and deterministic model of 1984. The method of Hopfield and Tank for mapping a combinatorial optimization problem onto a Hopfield network is then described, using the TSP as an example (Hopfield and Tank, 1985). The section continues with a discussion of the criticisms of the approach. We then briefly review some of the many modifications and extensions that have been made to the original model and approach in an attempt to overcome these limitations, including stochastic and chaotic variations. Finally, the reported performance of these Hopfield network models for combinatorial optimization across a range of benchmarked problems is discussed and compared to other heuristic approaches.

2.1. Discrete and stochastic Hopfield network

The original Hopfield network, as described in Hopfield (1982) comprises a fully interconnected system of n computational elements or neurons. In the following description, Hopfield's original notation has been altered where necessary for consistency. The strength of the connection, or weight, between neuron i and neuron j is determined by W_{ij}. This weight may be positive or negative depending on whether the neurons act in an excitatory or inhibitory manner (or zero if there is no interaction). Each neuron has an internal state u_i and an external state v_i. While the internal states are continuously valued, the external states are binary for this discrete model. The relationship between the internal and external states of the neurons can be shown as:

$$u_i(t+1) = \sum_{j=1}^{n} W_{ij} v_j(t) + I_i \tag{1}$$

$$v_i(t+1) = f(u_i) = \begin{cases} 1 & \text{if } u_i > 0 \\ 0 & \text{if } u_i \leq 0 \end{cases} \tag{2}$$

where I_i is a constant external input to neuron i and $f(\,)$ is the transfer function between internal and external states. The connection weights $\mathbf{W}$ are also constant, and the only variable elements in the network are the internal and external states of the neurons that are updated over time. From equations (1) and (2) it is clear that the internal state of each neuron is calculated as the weighted sum of inputs from its connected neurons, with an additional constant input. The neuron will "fire" (as evidenced by an external state of 1), if it receives sufficient stimulation from its connecting neurons, otherwise the neuron's external state will be zero representing a dormant or "non-firing" state.

The neurons update themselves over time in a random sequence, thus the model is said to be discrete and stochastic. As the network updates, and provided the weight matrix is symmetric with non-negative diagonals, the following

energy function is guaranteed to be minimized until the system converges to one of its stable states.

$$E_d = -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} W_{ij}v_i v_j - \sum_{i=1}^{n} I_i v_i. \tag{3}$$

Using the terminology of operations research, the system of equations (1) and (2) performs a gradient descent on the energy function, with the neuron states $\mathbf{v}$ converging to one of its local minima. If the values of the weights $\mathbf{W}$ and external inputs $\mathbf{I}$ are fixed appropriately, this process can be used to minimize any quadratic function of binary variables.

2.2. Continuous and deterministic Hopfield network

Hopfield's subsequent modifications to the original 1982 model were driven by considerations of biological plausibility. In the biological system, u_i lags behind the instantaneous outputs v_j of the other neurons because of the input capacitance C_i of the cell membrane, the trans-membrane resistance R_i, and the finite impedance $R_{ij} = W_{ij}^{-1}$ between the output v_j and the cell body of neuron i. The external states of the neurons are now continuous valued between 0 and 1, rather than binary, as in the earlier model, and represent an average "firing rate". Hopfield (1984) modeled this more biologically based system using the following resistance-capacitance differential equation to determine the rate of change of u_i, and hence the time evolution of the continuous Hopfield network:

$$\frac{du_i}{dt} = \sum_{j=1}^{n} W_{ij}v_j - \frac{u_i}{\tau} + I_i \tag{4}$$

$$v_i = f(u_i) \quad \text{and } \tau = R_i C_i \tag{5}$$

where the transfer function $f(\,)$ is now a continuous sigmoidal function such as:

$$v_i = f(u_i) = \frac{1}{2}\left(1 + \tanh\left(\frac{u_i}{T}\right)\right) \tag{6}$$

and T is a parameter used to control the slope of the transfer function. τ is the value of the time constant of the amplifiers, and without loss of generality can be assigned the value of unity, provided the time step of any discrete-time simulation of equation (4) is considerably smaller than unity. This same set of equations represents a resistively connected network of electronic amplifiers, and thus the system can be implemented with electrical circuits. We refer the interested reader to Hopfield (1984) for details of this implementation.

Similar to the original discrete model, the dynamics of this continuous model also minimizes an energy function over time, guaranteed to converge to stable states. This energy function is:

$$E_c = -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} W_{ij}v_i v_j - \sum_{i=1}^{n} I_i v_i + \int_0^{v_i} f_i^{-1}(\mathbf{v})d\mathbf{v}. \tag{7}$$

Hopfield (1984) showed that provided the weight matrix is symmetric, this function is a Liapunov function for the system of equations (4) and (5). Furthermore, if the slope of the transfer function (6) is particularly high (e.g. T is near zero), then the transfer function (6) approximates the behavior of the discrete version given by equation (2), and the integral term of expression (7) vanishes. Consequently, the local minima of E_c coincide with the local minima of E_d, and all these local minima lie at the vertices of the unit hypercube resulting in binary values for v. Thus, for T near zero, the continuous Hopfield network converges to a 0–1 solution in v which minimizes the energy function E_d.

Thus, there are two Hopfield neural network models available: a discrete version and a continuous version. The continuous version can either be implemented using electrical circuits, or simulated on a digital computer using an approximation to the differential equation (4) such as the Euler approximation. The accuracy of this approximation depends on parameters like the time step of the discretization, and affects the degree to which the discretized dynamics converge on the Liapunov energy function. Clearly, a small time step will approximate the dynamics well, ensuring gradient descent on the energy function.

2.3. Adaptation to solve combinatorial optimization problems

In 1985, John Hopfield teamed together with David Tank to extend the applications of his model to include solving combinatorial optimization problems (Hopfield and Tank, 1985). Hopfield and Tank (H–T) realized that networks of neurons with this basic organization could be used to compute solutions to specific optimization problems by selecting weights and external inputs which appropriately represent the function to be minimized and the desired states of the problem. The updating of the neurons according to the differential equations given by (4) and (5) (or even the discrete versions (1) and (2)) ensures that both the energy function and the optimization problem are simultaneously minimized over time. The analog nature of the neurons and the hardware implementation of the updating procedure could be combined to create a rapid and powerful solution technique.

Using the method proposed by Hopfield and Tank, the network energy function is made equivalent to the objective function to be minimized, while the constraints of the problem are included in the energy function as penalty terms.

Consider the quadratic formulation of the N-city Traveling Salesman Problem, given the binary decision variable

$$X_{ij} = \begin{cases} 1 & \text{if city } i \text{ is in position } j \\ 0 & \text{otherwise} \end{cases}$$

and the constant distance matrix d_{ik} representing the distance between cities i

and k:

$$\text{minimise} \quad \sum_{i=1}^{N} \sum_{\substack{k=1 \\ k \neq i}}^{N} \sum_{j=1}^{N} d_{ik} X_{ij} (X_{k,i+1} + X_{k,i-1}) \tag{8}$$

$$\text{subject to} \quad \sum_{i=1}^{N} X_{ij} = 1 \quad \text{for all } j \tag{9}$$

$$\sum_{j=1}^{N} X_{ij} = 1 \quad \text{for all } i \tag{10}$$

$$X_{ij} \in \{0, 1\} \quad \text{for all } i, j. \tag{11}$$

Apart from being a well-benchmarked problem, the TSP is a useful problem to consider since its form is that of a quadratic assignment problem. Thus the methods used by Hopfield and Tank for mapping the optimization problem onto a Hopfield neural network can be generalized to a wide range of problems with similar constraints and objective types.

The first step is to construct an energy function representation of the complete optimization problem using a penalty parameter approach, so that all objective functions and constraints are integrated into a single function which needs to be minimized. This is achieved by observing that a constraint of the form (9) can be enforced by ensuring minimization of the quantity

$$\sum_{i=1}^{N} \sum_{\substack{k=1 \\ k \neq i}}^{N} X_{ij} X_{kj} \quad \text{for all } j.$$

That is, a constraint requiring a single "1" in each column can be enforced by minimizing the pairwise product of elements in each column. If there is no more than one "1" in the column, then this term will be at its minimum value of zero. If there is more than one "1" in the column, then this term will be greater than zero. A similar term can be constructed to enforce the row constraint (10). Naturally, these terms will also be zero if there are no "1"s in each row or column as well. Since we need exactly one "1" per column and row, we will also need an additional term to force N elements of the solution matrix $\mathbf{X}$ to be "1"s.

The complete set of constraints can therefore be enforced through minimization of penalty terms, and when we add the objective function to these terms, we arrive at the H–T energy function for the TSP:

$$E = \frac{A}{2} \sum_{j=1}^{N} \sum_{i=1}^{N} \sum_{\substack{k=1 \\ k \neq i}}^{N} X_{ij} X_{kj} + \frac{B}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{\substack{l=1 \\ l \neq j}}^{N} X_{ij} X_{il}$$

$$+\frac{C}{2}\Big(\sum_{i=1}^{N}\sum_{j=1}^{N}X_{ij}-N\Big)^2+\frac{D}{2}\sum_{i=1}^{N}\sum_{\substack{k=1\\k\neq i}}^{N}\sum_{j=1}^{N}d_{ik}X_{ij}(X_{k,i+1}+X_{k,i-1}) \qquad (12)$$

The first two terms enforce no more than one "1" per column and row, respectively, the third term ensures that there are N elements "on" in the solution matrix, and the final term minimizes the tour length. The penalty parameters A, B, C and D need to be fixed at values that reflect the relative importance of these terms in the minimization process. If A, B and C are not large enough relative to D, then the resulting solution may be infeasible. Similarly, if D is not large enough, the solution may be feasible but the tour length may be larger than the optimal value. Hopfield and Tank (1985) used values of $A = B = D = 500$ and $C = 200$ to balance these terms.

Now that the energy function has been constructed, the next step is to derive the Hopfield network weights and external inputs so that the energy function is minimized by the network dynamics. For this we need to expand and rearrange the energy function (12) so that it is in the same form as the standard Hopfield energy function E_d

$$E_d=-\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{N}\sum_{l=1}^{N}W_{ijkl}X_{ij}X_{kl}-\sum_{i=1}^{N}\sum_{j=1}^{N}I_{ij}X_{ij} \qquad (13)$$

which has been modified to reflect the fact that the neurons X_{ij} for our TSP problem are two dimensional, compared to the linear array of neurons vi used in the standard Hopfield network. Once the forms of these two functions (12) and (13) are similar, the network weights $\mathbf{W}$ and external inputs $\mathbf{I}$ can be read as the coefficients of the quadratic and linear terms respectively. To ensure equivalence of the two functions, the summations of each term in (12) need to be extended across all relevant dimensions (i, j, k, l for quadratic terms and i, j for linear terms). Thus the Kronecker–Delta symbol is incorporated into each term of (12) where necessary:

$$\delta_{ab}=\begin{cases}1 & \text{if } a=b\\ 0 & \text{if } a\neq b.\end{cases}$$

Expanding (12) and rearranging the terms into quadratic, linear, and constant terms, thus yields:

$$\begin{aligned}E=&-\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{N}\sum_{l=1}^{N}[-A\delta_{ik}(1-\delta_{jl})-B\delta_{jl}(1-\delta_{ik})-C\\ &-D\delta_{ik}(\delta_{l,j+1}+\delta_{l,j-1})]X_{ij}X_{kl}-\sum_{i=1}^{N}\sum_{j=1}^{N}[CN]X_{ij}+\frac{CN^2}{2}\end{aligned} \qquad (14)$$

By comparing (14) to the standard Hopfield energy function (13) we obtain that the network parameters are:

$$\begin{aligned} W_{ijkl} &= -A\delta_{ik}(1-\delta_{jl}) - B\delta_{jl}(1-\delta_{ik}) - C - D\delta_{ik}(\delta_{l,j+1}+\delta_{l,j-1}) \\ I_{ij} &= CN. \end{aligned} \tag{15}$$

The constant term in equation (14) can be ignored since it merely reflects a shift upwards and does not affect the location of the minima of the function.

Now that the network weights and external inputs are determined, the Hopfield network can be initialized (using random values for the initial states), and updated according to equations (4) and (5) (or equations (1) and (2) for a purely discrete version). This updating is guaranteed to minimize the Hopfield energy function (13), and since this function is equivalent to the TSP energy function (12) and (14), then the resulting solution matrix $\mathbf{X}$ will provide a local minima of the TSP energy function. The quality and feasibility of this solution depends on the choice of penalty parameters A, B, C and D, as well as the initialization of the neurons, and the accuracy with which the dynamics of the differential equation (4) can be simulated if the continuous model is chosen.

The complete, the procedure is summarized in the pseudocode form below:

```
Step 0: Preliminary Tasks

        0.1 Construct an energy function for the optimization
            problem using a penalty parameter approach

        0.2 Expand energy function and infer network weights and
            external inputs

Step 1: Initialization Tasks

        1.1 Initialize neuron states to random values

        1.2 Select A, B, C, D

        1.3 Select T, the parameter of the continuous transfer
            function, and the value of the discrete time step if
            simulating the continuous model


Step 2: If energy function has converged to local minimum proceed
        to Step 5, otherwise proceed to Step 3

Step 3: Repeat n times:

        3.1 Randomly choose a neuron i to update (if using
            discrete time dynamics)

        3.2 Update u_i and v_i using equations (1)--(2) or (3)--(4)
```

Step 4: Go back to Step 2

Step 5: Examine final solution matrix and determine feasibility and optimality

Step 6: Adjust parameters A, B, C, D if necessary to obtain a satisfactory solution, re-initialize neuron states, and repeat from Step 2

Clearly, one of the main limitations of the H–T approach to solving combinatorial optimization is the difficulty in choosing appropriate penalty parameters. In addition to this difficulty, the Hopfield network performs a gradient descent on the energy function, and thus converges to the first local minimum encountered. Coupling these two issues, it seems likely that the H–T approach may yield solutions of poor quality. Wilson and Pawley (1988) first published these findings nearly three years after Hopfield and Tank's original paper was published. In doing so, they raised serious doubts as to the validity of the H–T approach to solving optimization problems, which seemingly served to dampen the enthusiasm surrounding the field.

2.4. Extensions

Since Wilson and Pawley's results were published, it has been widely recognized that the H–T formulation is not ideal, even for problems other than the TSP. The problem of optimal selection of the penalty parameters is not trivial and much work has been done to try to facilitate this process (Hedge et al., 1988; Kamgar–Parsi, 1992; Lai and Coghill, 1992). Many other researchers believed that the H–T energy function needed to be modified before any progress would be made, and considerable effort has also been spent in this area (Brandt et al., 1988; Van den Bout and Miller, 1988). One obvious improvement to the H–T approach to solving the TSP is to reduce the number of terms needed to represent the constraints by using the form $(\sum_{i=1}^{N} X_{ij} - 1)^2$ to represent the column constraints, for example. This eliminates the need for the third term in equation (12), so that the penalty parameter C is also eliminated.

Perhaps the most important breakthrough in the field, however, came from the valid subspace approach of Aiyer et al. (1990), and the subsequent work of Gee (1993). Their idea was to represent the constraint set as a hyperplane, and encourage the solution to lie upon it. This is achieved by including a single term in the energy function for the constraints which attempts to minimize the deviation between the solution matrix and the constraint plane, or valid subspace. A single penalty parameter needs to be selected, which if large enough, will guarantee the feasibility of the final solution.

Some researchers have also attempted to address the limitations of the H–T approach by considering alternative representations of constraints, suitable values for penalty parameters, and other modeling issues. The majority of other

researchers in the field, however, have focused on the limitation of the Hopfield network dynamics. By extending the network dynamics to include stochasticity and hill-climbing capabilities, various methods have emerged that attempt to avoid the many local minima of the energy function.

The variations of the Hopfield network that have been proposed can be broadly categorized as deterministic, stochastic, or chaotic. The deterministic approaches include problem specific enhancements such as the "divide and conquer" method of Foo and Szu (1989) for solving the TSP, deterministic hill-climbing such as the "rock and roll" perturbation method of Lo (1992), and the use of alternative neuron models within the Hopfield network such as the winner-take-all neurons used by Amartur et al. (1992) to improve the feasibility of the solutions. Stochastic and chaotic approaches address the problem of poor solution quality by attempting to escape from local minima, and are briefly reviewed in the following sections.

2.4.1. Stochastic models

There are basically four main methods found in the literature to embed stochasticity into the Hopfield network:

1. replace sigmoidal transfer function with a stochastic decision-type function
2. add noise to the weights of the network
3. add noise to the external inputs of the network
4. any combination of the above methods.

The *Boltzmann machine* (Aarts and Korst, 1989; Hinton et al., 1984) utilizes the first method based on a discrete Hopfield model. The inputs are fixed, but the discrete transfer function is modified to become probabilistic. Much like simulated annealing (Kirkpatrick et al., 1983), the consequence of modifying the binary transfer level of each neuron is evaluated according to the criteria of the Boltzmann probability factor. This model is able to escape from local minima, but suffers from extremely large computation times. In order to improve the efficiency and speed of the Boltzmann machine, Akiyama et al. (1989) proposed Gaussian machines which combine features of continuous Hopfield networks and the Boltzmann machine. Gaussian machines have continuous outputs with a deterministic transfer function like the Hopfield network, but random noise is added to the external input of each neuron. This noise is normally distributed (or Gaussian) with a mean of zero and a variance controlled by a temperature parameter T. However, based upon Szu's fast simulated annealing (Szu and Hartley, 1987) which uses Cauchy noise to generate new search states and requires only a $T/\log(T)$ cooling schedule, the Cauchy machine (Szu, 1988; Takefuji and Szu, 1989) was proposed to improve solution quality. The Cauchy distribution is thought to yield a better chance of convergence to the global minimum than the Gaussian distribution. Furthermore, Cauchy noise

produces both local random walks and larger random leaps in solution space, whereas Gaussian noise produces only local random walks (Takefuji and Szu, 1989). The noise is incorporated into the transfer function, while the outputs of the Cauchy machine are binary. In the high-gain limit of the stochastic transfer function (T near zero), the Cauchy machine approaches the behavior of the discrete (and deterministic) Hopfield network. Another stochastic approach, which has been very successful is mean-field annealing (Peterson and Soderberg, 1989; Van den Bout and Miller, 1989, 1990), so named because the model computes the mean activation levels of the stochastic binary Boltzmann machine.

2.4.2. Chaotic models

Recently, chaotic neural networks (CNN) exploiting the rich behaviors of nonlinear dynamics have been developed as a new approach to extend the problem solving ability of standard Hopfield neural networks (Aihara, 1994). Aihara et al. (1990) first proposed a general neuron model with chaotic dynamics, which constituted a CNN that encompassed various associative and back-propagation networks. The model was applied to solve the TSP with higher efficiency and better solution quality than the traditional Hopfield network (Aihara et al., 1990; Yamada et al., 1993). Although chaotic dynamics was found to improve optimization, the unstable neuron outputs can be difficult to interpret, and a convergent network is more desirable for practical purposes. To meet both ends, a deterministic simulated annealing algorithm was proposed by Chen & Aihara (1995), where the self-feedback strength W_{ii} acts as the bifurcation parameter of the network dynamics. As the parameter is gradually reduced, a reverse bifurcation process known as chaotic simulated annealing (CSA) results, in contrast with stochastic simulated annealing (Kirkpatrick et al., 1983). The process starts with an unstable phase for searching global minima, followed by a stable, convergent phase. To show the effectiveness of the algorithm, the TSP, a maintenance scheduling problem (Chen & Aihara, 1995) and the N-queen problem (Kwok et al., 1998a) were computationally solved with high efficiency and solution quality. Methods combining conventional heuristic techniques like the 2-opt algorithm, tabu search, etc. with CNN were investigated by Hasegawa et al. (1997a; 1998) with good performance even on larger problems. To better understand the theoretical aspects of CNN's, the existence of chaotic structure and stability of discrete-time neural networks underlying the CSA algorithm was proved by Chen & Aihara (1997), and a dynamical mechanism explaining the efficiency and novel properties of CNN for optimization was described by Tokuda et al. (1997). This research suggests a crisis-induced intermittent switching phenomenon to be the dynamical mechanism of chaotic search for minima in the Hopfield energy landscape.

Another general approach to CNN involves Euler discretization of the continuous Hopfield network. In the framework of globally coupled map (GCM) dynamics, Nozawa (1992) proposed an alternative approach to construct a CNN by

using Euler discretization on the Hopfield network with negative self-couplings, which was equivalent to the simplest version of Aihara et al.'s model (1990). The TSP was solved computationally with this GCM model, and was found to be more efficient than the Hopfield network and some stochastic networks like the Boltzmann machine and Gaussian machine. More recently, Wang and Smith (1998) proposed a CSA scheme with the time step of the Euler discretization as the bifurcation parameter that controls the reverse bifurcation process. This provides chaotic minima searching as well as convergence to a stable solution. For some parameter values, this model is equivalent to the one proposed by Chen and Aihara (1995) but the underlying chaotic mechanism is quite different.

The ability of various CNN models to improve optimization raises questions as to which chaotic properties most benefit optimization performance and how they arebrought about. One way to approach the problem is to add external noise, correlated or not, into the network, and compare its response and optimization performance with CNN's that generate chaotic dynamics internally. Hayakawa et al. (1995) compared the effects of adding random noise and the logistic map time series into the Euler discretized Hopfield network, and found that short time correlation of the chaotic time series is effective for the search of global minima. Asai et al. (1995) also experimentally studied how autocorrelation in various chaotic time series improves the tracing of optimal solutions when solving the TSP. A more detailed study along this line by Hasegawa *et al.* (1997b) added surrogates of the logistic map time series to a Hopfield-like network and compared the TSP optimization performance to using random and $1/f^a$ noise. Other research includes using the Henon map time series as noise (Zhou et al., 1997), and solving the N-queen problem with added logistic map noise (Kwok et al., 1998b). It should be noted that although various methods of generating chaotic noise have been found to improve optimization, there is no strong evidence of random noise being less effective in general. Also lacking is a detailed account of the underlying mechanism for chaotic noise to improve optimization performance.

From the brief outline given above, we can see that there are currently two major classes of CNN. One is the internal approach, where chaotic dynamics is generated within the network controlled by some bifurcation parameters. Examples of this type include Chen and Aihara's decaying self-feedback CSA, Wang and Smith's decaying timestep CSA, and Nozawa's GCM model. The other class contains CNN models employing an *external* approach, where an externally generated chaotic signal is added to the network as perturbation. All CNN's utilizing externally generated chaotic noise belong to this class. To seek a better understanding of the functional aspects of chaotic dynamics existing in various CNN's, a unified framework was recently proposed (Kwok & Smith, 1999a). It allows us to compare and highlight important common features among the many CNN models, as well as to draw new classifications and insights, thereby providing a basis for constructing new models.

2.5. Comparison of gradient-based neural etworks with other heuristics

Hopfield and Tank successfully applied their approach to several optimization problems including an analog-to-digital converter, a signal decision circuit, and a linear programming model (Tank and Hopfield, 1986). It was, however, their results for the combinatorial TSP that attracted the most attention. Hopfield and Tank (1985) simulated a network of 10 cities (100 neurons), chosen at random on the interior of a 2-dimensional unit square. Their results for small-sized problems were quite encouraging. For a 10-city problem, and for 20 random starts, 16 converged to valid tours. About 50% of the trials produced one of the two known shortest tours. Hopfield and Tank then studied a 30 city (900 neuron) problem. Since the time required to simulate the differential equations on a computer scales worse than $O(n^3)$, their results were fragmentary. They were unable to find appropriate penalty parameters to generate valid tours, and commented that "parameter choice seems to be a more delicate issue with 900 neurons than with 100". In fact, their best solution was around 40% away from the best known solution of Lin and Kernighan (1973) on the same 30-city problem.

Since then, the many modifications to the original H–T approach have seen considerable improvement in these results. A recent fuzzy modification of Aiyer's subspace approach yielded nearest-city quality tours for up to 100 randomly generated cities (Wolfe, 1999). Peterson and Soderberg reported solutions for 200 cities using a mean field annealing neural network that were only slightly worse than simulated annealing results (Peterson and Soderberg, 1993). These results are still a long way from those that can be obtained by well-known heuristics. For example, the iterated Lin-Kernighan heuristic can routinely find solutions within 1% of optimal for problems with thousands of cities (Johnson, 1990). Even other neural network approaches such as the deformable template methods discussed in the next section yield considerably better results than the Hopfield variations seem capable of.

The advantage of the H–T approach to combinatorial optimization, however, lies in its generalization abilities. The H–T approach can be applied to any combinatorial optimization problem that can be formulated within quadratic terms. It does not rely on the geometry of the problem like many of the TSP heuristics or the deformable template methods. The many variations of the Hopfield network that have emerged over the last decade or so have been applied to a wide range of classical combinatorial optimization problems including assignment problems, constraint satisfaction problems, graph problems, integer programming, and scheduling problems to name a few. We refer the interested reader to Smith (1999) for a survey of these and other applications. Many of the results are competitive with other meta-heuristic approaches. One of the deficiencies of the literature in this area, however, is the fact that few studies are established as comparative analyses, aimed to determine the competitive-

ness of the proposed neural network approach with the best known heuristic or meta-heuristic approaches to the same problem. This makes a true evaluation of the performance of Hopfield-type models difficult. As Looi (1992) noted, "although there is a large collection of operations research based and other methods for solving all of these problems, comparisons between neural network methods with existing methods have been lacking". Solutions to this problem in the form of guidelines for experiments have now been published (Barr et al., 1995; Hooker, 1995) and we hope that researchers will soon provide enough studies of this nature, so that an accurate evaluation of the performance and potential of Hopfield-type neural networks on a wide variety of problems can be established.

3. Deformable template neural networks

Elastic nets (EN) and Self-organizing maps (SOM), often referred to as deformable templates, provide alternatives for solving low-dimensional problems with a geometric interpretation, like the Euclidean TSP. These models are fundamentally different from the H–T approach, as they evolve in a low-dimensional continuous search space. In the following, we describe both models for solving the Euclidean TSP. We then establish some relationships between the two models and present a few extensions, including a generalized version of the self-organizing map that is suitable for non-geometric problems.

3.1. Elastic net

The elastic net (EN) of Durbin and Willshaw (1987), originated from a previous work by Willshaw and von der Malsburg (1979). It is an iterative procedure where M points, with M typically larger than the number of vertices (or cities) N, are lying on a circular ring or "rubber band" originally located at the center of gravity of the vertices. The rubber band is gradually elongated until it is sufficiently close to each vertex to define a tour. During that process two forces apply: one for minimizing the length of the ring, and the other for minimizing the distance between the vertices and the ring. These forces are gradually adjusted as the procedure evolves. Figures 1(a), 1(b) and 1(c) show how the elastic net typically evolves over time. In the figure, the small black circles are the points located on the ring which are migrating towards the vertices in the Euclidean plane. When there is a point on the ring sufficiently close to each vertex, a solution is obtained, as shown in Figure 1(d).

This model will now be presented more formally, using a pseudocode notation. Let X_i be the coordinates of vertex i, $i = 1, \ldots, N$, Y_j the coordinates of ring point j, $j = 1, \ldots, M$, and $d_{X_i Y_j}$ the Euclidean distance between i and j. We have:

Step 0: $K \leftarrow K_0$; $Y_j \leftarrow Y_j^0$, $j = 1, \ldots, M$;

Step 1: Repeat rep times

1.1 Update the coordinates Y_j of ring point j, $j = 1, \ldots, M$

1.2 If $\min_{1,\ldots,M} d_{X_i Y_j} \leq \varepsilon$, $i = 1, \ldots, N$, then STOP

Step 2: $K \leftarrow \alpha K$ $(0 < \alpha < 1)$

Step 3: Go back to Step 1

Step 0 initializes the scale parameter K (see below) and selects an initial location for the points on the ring. In Step 1, the points migrate towards the vertices through an iterative procedure governed by parameter K. After a fixed number of iterations, related to the size of the problem, the value of parameter K is slightly reduced and the migration process is pursued with this new value. This is repeated until either K becomes smaller than some preset $K_{\min}$ value or there is a point on the ring sufficiently close to each vertex, as specified by the tolerance ε. Parameter K is reminiscent of the temperature parameter in the simulated annealing algorithm, as its value must be progressively reduced according to a pre-specified "cooling schedule" to obtain a good solution to the problem. An alternative or additional stopping criterion for EN is to iterate until some preset $K_{\min}$ value is reached and then, to associate each vertex with the closest ring point.

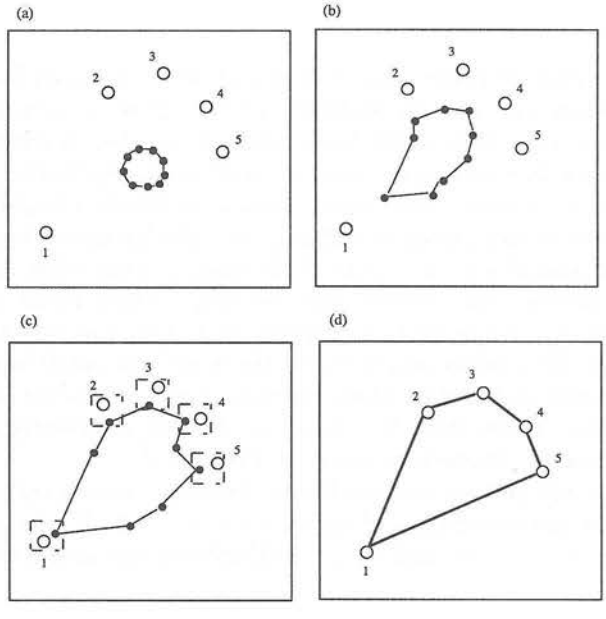

Figure 1. Evolution of the elastic net over time (a) (b) (c) and the final tour 1-2-3-4-5 (d).

In Step 1.1, the coordinates of each ring point j are updated as follows:

$$Y_j \leftarrow Y_j + \Delta Y_j$$

where

$$\Delta Y_j = \alpha \sum_{i=1,\ldots,N} w_{ij}(X_i - Y_j) + \beta K(Y_{j+1} + Y_{j-1} - 2Y_j) \tag{16}$$

$$w_{ij} = \frac{\phi(d_{X_i Y_j}, K)}{\sum_{k=1,\ldots,M} \phi(d_{X_i Y_k}, K)}, \quad i = 1, \ldots, N \tag{17}$$

$$\phi(d, K) = e^{\frac{-d^2}{2K^2}} \tag{18}$$

where, α, β are constant parameters and wij is a normalized measure of the "attraction" of vertex i on ring point j. In equation (16), the α term drives the points on the ring towards the vertices, and the β term keeps neighboring points on the ring together during the migration process to produce a short tour (i.e. neighboring points are associated with vertices that are close in distance). These two forces are illustrated in Figure 2. Force (1), derived from the α term, drives point j towards vertex i. Force (2), derived from the β term, is more easily understood by considering the following equivalence

$$Y_{j+1} + Y_{j-1} - 2Y_j = (Y_{j+1} - Y_j) + (Y_{j-1} - Y_j) \tag{19}$$

It thus defines a tension on the ring that keeps neighboring points together. Through parameters α and β, the relative strength of the two forces can be regulated.

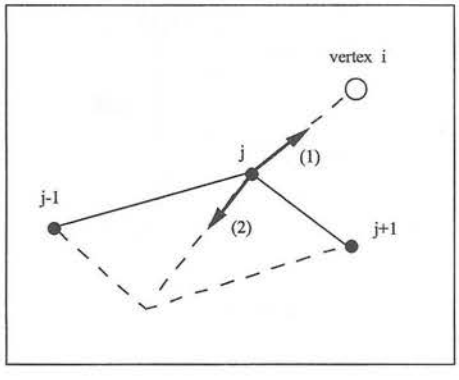

Figure 2. Forces that apply on ring point j.

It is worth noting that the update equations (16) can be expressed as the derivative of an appropriate energy function E_K, namely

$$\Delta Y_j = -K \frac{\partial E_K}{dY_j} \tag{20}$$

where

$$E_K = -\alpha K \sum_{i=1,\ldots,N} \ln \sum_{j=1,\ldots,M} \phi(d_{X_i Y_j}, K) + \frac{\beta}{2} \sum_{j=1,\ldots,M} d^2_{Y_j Y_{j+1}} \tag{21}$$

This algorithm thus finds a local minimum of such energy function by performing a gradient descent in a continuous two-dimensional Euclidean space. When K approaches 0 and the ratio of M to N approaches infinity, minimizing the energy is equivalent to minimizing the total length of the ring and, thus, the solution value. Since the shape of the energy function and its local minima change with K, the function is gradually modified through a slow reduction of the value of parameter K until the minima correspond to good TSP tours.

3.2. Self-organizing map

A self-organizing map is an instance of the so-called competitive neural networks (Kohonen, 1982, 1988). It is composed of a layer of input units fully connected to a layer of output units, the latter being organized according to a particular topology, such as a ring structure. Self-organizing maps basically produce topological mappings from high-dimensional input spaces to low-dimensional output spaces. In the case of a ring structure, the p-dimensional input vectors are associated with output units or 1-dimensional positions on the ring. The mapping is such that two input vectors that are close in the input space will be associated with units that are close on the ring.

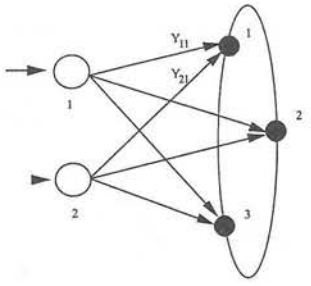

Figure 3. A SOM

In Figure 3, a SOM with $P = 2$ white input units and $M = 3$ black output units is shown. In this figure, Y_{11} and Y_{21} denote the weights on the connections

from the two input units to output unit 1, and $Y_1 = (Y_{11}, Y_{21})$ is the weight vector associated with output unit 1. In a TSP context, each input vector corresponds to the coordinates of a vertex. We then want vertices that are close in distance to be associated with units that are close on the ring to obtain a short tour.

This is obtained through the following iterative adjustment of the connection weights. Let us assume that we have a SOM with two input units and M output units on a ring, each with weight vector $Y_j = (Y_{1j}, Y_{2j})$, $j = 1, \ldots M$, $M \geq N$. Let $X_i = (x_i, y_i)$ be the coordinates of vertex i, $i = 1, \ldots N$. and $d_{X_i Y_j}$, the Euclidean distance between vertex i and output unit j. Then, we have:

`Step 0: Initialization.` $i \leftarrow 0;\ Y_j \leftarrow Y_j^0;\ j = 1, \ldots, M$

`Step 1: Competition.`

1.1 $i \leftarrow (i+1) \pmod{N+1}$

1.2 $o_j \leftarrow d_{X_i Y_j},\ j = 1, \ldots M$

1.3 $o_{j^*} \leftarrow \min_{j=1,\ldots M}\{o_j\}$

`Step 2: Weight adjustment.`

$$Y_j \leftarrow Y_j + \mu f(j, j^*)(X_i - Y_j), \qquad j = 1, \ldots, M,\ 0 < \mu < 1$$

`Step 3: If` $\min_{j=1,\ldots M} d_{X_i Y_j} \leq \varepsilon,\ i = 1, \ldots, N,$ `then STOP`

`Step 4: Go back to Step 1.`

In Step 1, the winning output unit j^* is the one with the closest weight vector to the current vertex in Euclidean distance. In Step 2, function f is typically a decreasing function of the lateral distance between output units j and j^* on the ring (i.e., if there are k units on the ring between the two, the lateral distance is $k+1$) and its range is the interval $[0, 1]$. Thus, the weight vector of the winning unit j^* and the weight vectors of units that are close to j^* on the ring all move towards the current vertex, but with decreasing intensity as the lateral distance to the winning unit increases. Typically, function f is modified as the algorithm unfolds to gradually reduce the magnitude of the weight adjustment. At the start, all units that are close to the winning unit on the ring "follow" that unit in order to move in the same area. At the end, only the weight vector of the winning unit significantly moves towards the current vertex.

This iterative adjustment procedure is repeated, through multiple passes over the set of vertices (see the modulo operator in Step 1.1) until there is a weight vector sufficiently close to each vertex. Other or additional stopping criteria may be considered, like a fixed number of passes through the vertices or stabilization of the competition, when it is observed that the winner for each vertex does not change from one pass to another. The association of each vertex with a weight vector (i.e., an output unit with an index or position on the ring)

produces a tour. If we consider the two-dimensional weight vector associated with an output unit on the ring as the location of that unit in Euclidean space, Figure 1 is still a good way of visualizing the evolution of the SOM with the output units on the ring migrating towards the vertices.

3.3. Extensions

Since both EN and SOM exploit the geometry of the problem, they have mostly been applied to the TSP. A few extensions are reported in the literature for the multiple TSP (Goldstein, 1990) and vehicle routing problems (Ghaziri, 1991, 1996; Matsuyama, 1991; Vakhutinsky and Golden, 1994; Potvin and Robillard, 1995). In these applications, multiple tours are formed through the migration in parallel of multiple rings. In the case of the vehicle routing problem, the capacity or time constraints typically break the geometrical nature of the problem and the algorithm must be modified accordingly. The SOM described in Ghaziri (1991), for example, involves competition for the current vertex at two different levels: one within each ring based on geometric properties and the other among the rings to take into account the capacity constraints.

3.4. Generalization to non-geometric problems

Some recent works have applied SOM concepts to non-geometric problems, like the generalized quadratic assignment problem (Smith, 1995). These approaches depart from the traditional ring structure and exploit more complex topologies. The concept of self-organization is embedded in a combinatorial framework rather than within a deformable template structure. This approach, known as the self-organizing neural network (SONN), was first proposed by Smith in 1995, and has been expanded to include measures for improving the optimization performance such as normalization and annealing (Guerrero et al., 1998; 1999; 2000; Lozano et al., 1998) and chaotic perturbation and analysis (Kwok and Smith, 2001).

The idea behind the SONN (Smith, 1995; Smith et al., 1996; 1998) acknowledges that the optimal solution to many combinatorial optimization problems involves finding the best location for rows of a permutation matrix, so that the solution is both feasible and minimizes the cost function. For example, a solution to the Travelling Salesman Problem may be represented as a permutation matrix

$$X = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \quad \text{where } X_{ij} = 1$$

if city j is visited in position i. Clearly, this solution indicates that the salesman visits cities in the order B, D, A, C. Each row (vector) corresponds to a particular

city, and we need to find the optimal arrangement of the four vectors so that the tour length is minimized (given the inter-city distances, etc.). The SONN views this problem as a competition: when presented with a given city via a vector (row of the permutation matrix), a competition takes place to determine the best row of the permutation matrix in which to place the city based on the objective function cost. This competition is conducted in a manner similar to the Kohonen self-organizing feature map (Kohonen, 1982), where a winning neuron (row number) is defined, and a neighborhood around the winner is also created. The architecture used is the same as Kohonen's: an input layer is connected via weights to a row of neurons. The input layer presents rows of the permutation matrix, while the output neurons correspond to the desired location (row of the permutation matrix) for the current input pattern. The weights are continuously valued, between zero and one, and indicate the degree to which each item (city in this example) belongs in each position. Over time, these weights approach binary states. Once an input is presented to the network, and the winning neuron has been determined, the weights connecting the input layer to the winning neuron and its neighbors are updated using a rule similar to Kohonen's to reflect the learning process. The main difference between the SONN and Kohonen's self-organizing process is in the definition of the neighborhood. The neighbors of the winning neuron are defined according to the cost function of the optimization problem, rather than the geographic location of the neurons. This difference is crucial, and has enabled the process of self-organization to be applied to a wide range of combinatorial optimization problems, thus escaping the previous limitation of the research to problems that could be embedded in a two- dimensional plane.

One of the known limitations of the SONN however, is its tendency to oscillate during the optimization process (Smith, 1995; Smith et al., 1996; 1998). Several methods have been used to overcome these oscillations including: adding a penalty term to the cost calculation to penalize too many contradictory weight changes (Smith et al, 1998); incorporating normalization and annealing (Guerrero et al., 1998; 2000); and adding a conscience mechanism that makes a neuron feel "guilty" if it is declared the winner too often (a common cause of oscillations) (Guerrero et al., 1999). Our recent research has investigated the role of chaos in helping to control the behavior of the SONN weight adaptations, and suggests some promising future directions (Kwok and Smith, 2001). Our analysis revealed that the SONN is extremely rich in nonlinear system dynamics due to interactions between the neurons and interactions between the weights of each neuron. For certain annealing schedules the SONN dynamics results in a period-doubling route to chaos, and the high feasibility bands correspond to the bifurcation regions. A strange attractor has also been found which matches the repeated folding and stretching of the state space during the weight updating and normalization procedure of the SONN (Kwok and Smith, 2001). We have therefore shown that the SONN contains significant hidden nonlinear system dynamics that play a critical role in the optimization performance of the model.

Exploiting these rich dynamics, much like chaos has been used to assist the performance of the Hopfield neural network, is therefore a promising research direction for this broadly applicable self-organizing approach.

3.5. Comparison of deformable templates with other heuristics

Both SOM and EN are closely related. They both involve migration of a ring towards vertices, although the mechanism for updating the position of the ring points is different. In the case of EN, the update is defined through the minimization of an appropriate energy function. There is no need to define such a function in the case of the SOM.

It is difficult to get an accurate picture of the comparative performance of the two models from the current literature. The results are scattered in different journals from different research areas. Computation times are often missing and comparisons with alternative methods are rare. Furthermore, the problems are not taken from standard benchmarks. Overall, it seems that both EN and SOM can find good quality solutions to medium-sized TSPs. For problems with up to a few hundred cities, EN often ends up with slightly better solutions (Angeniol et al., 1988). However, it is also more computationally expensive, as it requires more iterations to converge (see, e.g., the slow cooling schedule of scale parameter K). SOM scales up better and has been applied to much larger problems. In Favata and Walker (1991), for example, the authors report results on problems with up to 10,000 vertices. The solutions obtained were about 5% worse than those produced by a simulated annealing heuristic.

4. Comparison of neural approaches

While the previous sections presented a review of gradient-based and deformable template neural network models, this section aims to compare these approaches across a set of measures: efficiency, solution quality, and scope.

4.1. Efficiency

The efficiency of the Hopfield network for solving combinatorial optimization depends significantly on the way in which the problem is mapped onto the network. The encoding of the variables, constraints, and objective function into the Hopfield energy function and the values of the penalty parameters combine to determine the complexity of the energy surface. This, in turn, affects the degree to which the Hopfield network dynamics is able to search for local minima. It is interesting to note that almost all of the Hopfield network applications to the TSP have been based on the formulation used by Hopfield and Tank which uses a decision variable denoting if a city belongs to a given position in the tour sequence. The operations research community, however, has based many of its TSP methods on another formulation: here the decision variable X_{ij} denotes if

city i follows city j in the tour sequence. This alternative formulation results in a linear objective function, but some complex constraints are needed to avoid sub-tours. Hopfield and Tank's method cannot readily be applied to the linear formulation due to the difficulty of encoding this constraint (Smith et al., 1996). Nevertheless, this raises the issue of the existence of alternative formulations for a given problem, and the degree to which this impacts the efficiency of any resulting Hopfield networks. A recent study of a Hopfield network application to school timetabling has shown that where alternative formulations exist for a problem, the chosen formulation greatly impacts the dimensionality of the neurons, the number of constraints needed to represent the problem, and the complexity of the energy surface (Smith et al., 1999).

While the Hopfield network requires N^2 binary neurons to solve the TSP, the elastic net algorithm only requires $2M$ ring points, where M is usually selected to be around 2.5^*N (see Durbin and Willshaw, 1987). Updating the position of the ring points is computationally expensive, however, as the update of a single point depends on the position of every vertex through the attraction coefficients w_{ij}. Furthermore, these coefficients must be recomputed at each iteration. For $M \approx N$, the complexity of each iteration is thus $O(N^2)$. Different approaches have been proposed to reduce this burden. A natural way of addressing the problem without degrading too much solution quality is to consider only vertices that have a significant impact on the ring points (Boeres et al., 1992; Vakhutinsky and Golden, 1995). In other words, attraction coefficients that are not of sufficient magnitude, because their vertices are too far from the corresponding ring points, are filtered out. A large number of coefficients may be eliminated in this way because the function $\phi(d, K)$ decreases quickly as the square of the distance d^2 grows. Hierarchical elastic nets have also been proposed (Vakhutinsky and Golden, 1995) where the basic idea is to divide the area containing the vertices into smaller subareas and to replace the vertices in each subarea by a single vertex located at their center of gravity. As the algorithm unfolds, the subareas are progressively reduced until each subarea contains a single vertex. Working on smaller aggregated problems at the start allows the algorithm to find the general shape of the solution more quickly.

Improving the efficiency of the Self-Organizing Map has also been a concern of researchers. As noted by different authors (Burke and Damany, 1992; Favata and Walker, 1991), significant gains in efficiency are obtained by reducing the number of ring points that move towards the current vertex (i.e., those that are close to the winning point) and also by reducing the magnitude of the move. In Step 2 of the algorithm (see section 3.2), the update mechanism is governed by function $f(j, j^*)$ which has a form like:

$$f(j, j*) = \begin{cases} \left(1 - \dfrac{d''_{jj^*}}{L}\right)^{\beta}, & \text{if } d''_{jj*} < L, \\ 0, & \text{otherwise} \end{cases} \tag{22}$$

where $d''_{jj*} = \min(|j - j^*|, m - |j - j^*|)$.

In this definition, d'' is the lateral distance on the ring between point j and the winning point j^* (assuming that the points on the ring are indexed from 1 to M), and $|x|$ is the absolute value of x. This function is such that it always returns a value of 1 for the winning point, and this value decreases as the lateral distance from the winning point increases; when the lateral distance goes beyond parameter L, the points do not move at all. The value of parameter β is increased from one pass to another to progressively reduce the magnitude of the move of neighboring units. At the end, when β is sufficiently large, only the winning unit moves towards the current vertex and separates from its neighbors to fix the solution.

4.2. Solution quality

The traditional Hopfield neural network has difficulty in minimizing the competing terms in the energy function, and is well known for its tendency to yield infeasible or poor-quality solutions. As discussed in Section 2.4, there have been numerous attempts to overcome this limitation by either deriving alternative energy function forms to reduce the number of local minima, or by trying to balance the penalty factors. The choice of penalty factor values affects the contour of the energy function surface, and thus greatly affect the ability of the Hopfield network to find local minima of the optimization problem. For many types of constraints, the penalty factors can be treated as equivalent (e.g., penalty factors for row and column constraints in the TSP should be identical, since these constraints are equally important and equally difficult to satisfy). This observation can often reduce the search space for the optimal penalty factor combination. Many researchers have attempted to eliminate the need for trial and error parameter selection by examining the theoretical balancing of terms in the energy function. For the TSP, Hedge et al. (1988) showed that while some regions of parameter space can be identified that yield better quality results, the size of these regions diminishes rapidly as the problem size increases. Kamgar-Parsi and Kamgar-Parsi (1992) developed a systematic method for selecting the penalty factors based on analyzing the dynamical stability of feasible solutions. Trial and error searching, however, does not necessarily preclude a systematic method. The efficiency of searching for optimal penalty parameter values can be improved by adopting the following systematic approach: first find values for the penalty factors that provide a feasible solution, holding the objective function penalty factor constant at unity. Once a combination of penalty factors has been found that consistently yields feasible solutions, slowly start to increase the objective function factor in an attempt to produce less expensive feasible solutions. As soon as feasibility is lost, the bounds on this parameter can be established. This much reduced search space can then be explored in more detail to obtain the combination of penalty factors that yields consistently feasible and optimal solutions.

One of the factors that has made it difficult for researchers to reproduce Hopfield and Tank's original results is that they omitted certain critical details in their paper about the method used to simulate the differential equation, and the termination criteria. Wilson and Pawley (1988) experimented with three different termination criteria in an effort to reproduce the results: the network simulation was terminated if i) a valid tour was found, ii) the network had frozen as measured by no neuron v values changing by more than 10^{-35} since the last update, and iii) more than 1000 updating iterations had elapsed (a "time-out" test, useful for catching cyclic and oscillatory convergence). Wilson and Pawley found that 1000 iterations were sufficient for their experiments, and increasing the "time-out" range to 10,000 iterations did not produce any improvement in results. It is important to be aware, however, that the quality of the reported results is usually affected greatly by the termination criteria selected, and researchers need to be sure to report these accurately.

The solution quality of the elastic net is also highly dependent on optimal choice of many parameters. For example, the number of ring points M to be used is clearly related to the number of vertices N. However, using too many points leads to a loss of efficiency. In the literature, M is usually around $2.5N$. As noted by some authors (Favata and Walker, 1991), normalizing the coordinate vectors of the vertices can often lead to a more robust algorithmic behavior over different types of instances. The initial position of the ring can also affect the performance, and is typically around the center of gravity of the vertices. Good results are also reported when the points on the ring correspond to the convex hull of the vertex set (Burke, 1994). The parameter K also impacts the results greatly. When the value of parameter K is large, the energy function is rather smooth, but as this value is reduced a multimodal energy landscape emerges, where the good local minima should correspond to good tours. In order to obtain this result, parameter K must be slowly reduced to avoid some form of twisting or crossover of the ring (which typically leads to long tours). For example, a problem with 100 vertices was solved by Durbin and Willshaw (1987), by setting K to an initial value of 0.2 and by reducing it by 1% every 25 iterations until a value in the range of 0.01-0.02 was obtained.

The interpretation of the final configuration can sometimes impact the solution quality as much as the process of obtaining the final configuration, and needs to be carefully considered. Two or more different ring points may be associated with the same vertex if they are all within the tolerance of that vertex (this is sometimes referred to as a spike). Conversely, a ring point may be associated with two or more vertices. In such cases, the solution is not well defined. In Fig. 4(a), two ring points are within the tolerance of vertex 2, and two different sequences 1-2-3-4-5 and 1-3-4-2-5 are obtained depending of the ring point chosen. In Fig. 4(b), a single ring point is associated with vertices 2 and 3. Hence, it is impossible to know if vertex 2 is visited before or after vertex 3. One possible way of solving these problems is through post-processing. For example, all valid solutions obtainable with the current configuration can be considered

and the best overall solution is taken. Trying to avoid this phenomenon during the course of the algorithm is rather difficult. Some theoretical studies indicate that an appropriate setting of the β/α ratio (so that it is about one-half of the average inter-point distance on the ring) is likely to lead to fully-specified solutions (Simmen, 1991).

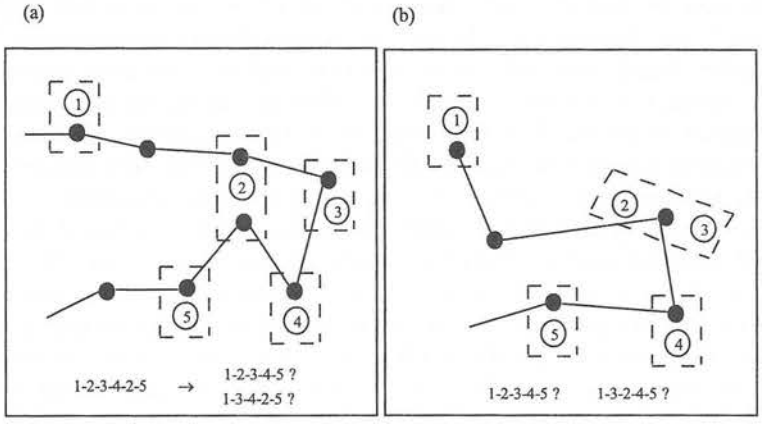

Figure 4. Solution ambiguity

Solution ambiguity can also occur in the SOM, either because many ring points fall within the tolerance of a given vertex or a single ring point falls within the tolerance of two or more vertices. However, due to the methodology adopted for stretching the ring, which is based on a competition between ring points, it may also happen that a number of points will freeze at their initial location, because they never win any competition. Consequently, partially defined tours are obtained, where a number of vertices are "orphans" (i.e., do not have any close ring points). Different techniques have been proposed to alleviate this problem.

i) In Angeniol et al. (1988) the implementation is based on the distinctive feature that ring points are dynamically created and deleted. A point is duplicated if it wins for two different vertices after a complete pass through the set of vertices. It is deleted, if it does not win after three complete passes. Through this mechanism, vertices are less likely to end up alone. Starting with a single point on the ring, the authors report that up to twice as many points as vertices may be created during the procedure.

ii) In Burke and Damany (1992), a conscience mechanism proposed by Desieno (1988) replaces the dynamic creation and deletion of ring points. A penalty is added to the distance between a ring point and a vertex,

based on the number of times that point has won the competition in the past. Consequently, frequent winners are heavily penalized in favor of other units. Basically, Step 1.2 (competition) of the algorithm presented in Section 3.2 is modified as follows for a given vertex i and ring point j:

Step 1.2'. $o_j \leftarrow d_{X_i Y_j} + \gamma b_j$,

where b_j is the penalty or bias associated with ring point j. This penalty is typically the fraction of competitions won by ring point j in the past. Good results are reported with a number of ring points now equal to the number of vertices, leading to substantial savings in computation time.

iii) Parameter γ that weighs the penalty with respect to the true distance in the conscience mechanism, is reportedly difficult to tune. In Burke (1994), a vigilant net is proposed where ring points are turned off if they win too often to let others win. Basically, the number of wins is recorded for each unit and that unit is turned off for the remaining part of the pass through the set of vertices, if this number exceeds some threshold value (known as the vigilant parameter). At the start of the next pass, the winning score of all units is reset to zero. The vigilance parameter is large initially, to allow the vertices to win freely at the start, and is progressively reduced until it reaches a value of one, to let the ring points separate and converge towards distinct vertices.

4.3. Scope

The Hopfield neural network and its many stochastic and chaotic variants find broad applicability across a wide range of combinatorial optimization problems provided the objective function and constraint terms are linear or quadratic. Fortunately, this covers a broad class of problems including the generalized quadratic assignment problem, a form in which many practical COPs can be expressed. The deformable template approaches of the elastic net and SOM however, are quite restricted in their scope and applicability: the problem must be geometric in nature as the solution evolves in a geometric plane. The self-organizing neural network (SONN) method of Smith, discussed in Section 3.4, extends the scope of the self-organizing principle to any combinatorial optimization problem, regardless of the nature of the objective function, although the constraints are such that the solution needs to be expressed in the form of a permutation matrix. Recently, other neural network methods, such as the back-propagation learning algorithm of feedforward neural networks, have been used to approximate complex objective functions using only a sample of data points (Smith and Gupta, 2001). The resulting neural model then represents a continuous and differentiable approximation to the original optimization problem, that can be optimized using any gradient-based or heuristic approach.

5. Conclusion

This paper has reviewed the two main types of neural network models that can be used for combinatorial optimization: gradient-based neural networks including the Hopfield networks and its many variants, and the deformable template models of elastic nets and self-organizing maps. The review has covered both the theoretical aspects of their application to combinatorial optimization problems, as well as discussed a variety of practical considerations that affect the performance of the models. The original models have been discussed along with deterministic, stochastic and chaotic variations designed to improve their performance.

From a meta-heuristics viewpoint, neural networks can be seen as an alternative technique with the current potential to match the performance of better known algorithms such as simulated annealing. This potential relies on due consideration of the aforementioned range of issues affecting the success and efficiency of the methods. The deformable template methods are well suited to solving low dimensional problems with geometric interpretation like the TSP. The Hopfield network method generalizes to a broad range of combinatorial problems, but the cost of this generalization is a reduction in efficiency and scalability. Certainly, current developments in hardware implementation of neural architectures should see some of these limitations relaxed in future years. The advantage of neural networks over other meta-heuristics could then be more fully determined.

References

AARTS, E.H.L. and KORST, J. (1989) *Simulated Annealing and Boltzmann Machines.* John Wiley & Sons, Essex.

AIHARA, K., TAKABE, T. and TOYODA, M. (1990) Chaotic neural networks. *Physics Letters A*, **144**, 333–340.

AIHARA, K. (1994) Chaos in neural response and dynamical neural network models: towards a new generation of analog computing. In: M. Yamaguti, *Towards the Harnessing of Chaos.* Elsevier Science Publishers B. V., Amsterdam, 83–98.

AIYER, S.V.B., NIRANJAN, M. and FALLSIDE, F. (1990) A theoretical investigation into the performance of the Hopfield model. *IEEE Transactions on Neural Networks*, **1**, 204–215.

AKIYAMA, Y., YAMASHITA, A., KAJIURA, M. and AISO, H. (1989) Combinatorial Optimization with Gaussian Machines. *Proceedings IEEE International Joint Conference on Neural Networks*, **1**, 533–540.

AMARTUR, S. C., PIRAINO, D. and TAKEFUJI, Y. (1992) Optimization Neural Networks for the Segmentation of Magnetic Resonance Images. *IEEE Transactions on Medical Imaging*, **11**, 215–220.

ANGENIOL, B., VAUBOIS, G. and LE TEXIER, J.Y. (1988) Self-Organizing Fea-

ture Maps and the Travelling Salesman Problem. *Neural Networks*, **1**, 289–293.

ASAI, H., ONODERA, K., KAMIO, T. and NINOMIYA, H. (1995) A study of Hopfield neural networks with external noises. *Proceedings of the IEEE International Conference on Neural Networks*, **4**, 1584–1589.

BARR, R.S., GOLDEN, B.L., KELLY, J.P., RESENDE, M.G.C. and STEWART, W.R. (1995) Designing and Reporting on Computational Experiments with Heuristic Methods. *Journal of Heuristics*, **1**, 9–32.

BOERES, M.S.C., DE CARVALHO, L.A.V. and BARBOSA, V.C. (1992) A Faster Elastic Net Algorithm for the Traveling Salesman Problem. *Proceedings of the Int. Joint Conf. on Neural Networks*, **2**, 215–220.

BRANDT, R.D., WANG, Y., LAUB, A.J. and MITRA, S.K. (1988) Alternative networks for Solving the Travelling Salesman Problem and the List-Matching Problem. *Proceedings International Conference on Neural Networks*, **2**, 333–340.

BURKE, L.I. (1994) Adaptive Neural Networks for the Traveling Salesman Problem: Insights from Operations Research. *Neural Networks*, **7**, 681–690.

BURKE, L.I. and DAMANY, P. (1992) The Guilty Net for the Traveling Salesman Problem. *Computers & Operations Research*, **19**, 255–265.

BURKE, L.I. and IGNIZIO, J.P. (1992) Neural Networks and Operations Research: An Overview. *Computers & Operations Research*, **19**, 179–189.

CHEN, L. and AIHARA, K. (1995) Chaotic Simulated Annealing by a Neural Network Model with Transient Chaos. *Neural Networks*, **8**, 915–930.

DESIENO, D. (1988) Adding a Conscience Mechanism to Competitive Learning. *Proceedings of the IEEE Int. Conf. on Neural Networks*, **1**, 117–124.

DURBIN, R. and WILLSHAW, D.J. (1987) An Analogue Approach to the Traveling Salesman Problem using an Elastic Net Method. *Nature*, **326**, 689–691.

FAVATA, F. and WALKER, R. (1991) A Study of the Application of Kohonen-type Neural Networks to the Traveling Salesman Problem. *Biological Cybernetics*, **64**, 463–468.

FOO, Y.P.S. and SZU, H. (1989) Solving large-scale optimization problems by divide-and-conquer neural networks. *Proceedings IEEE International Joint Conference on Neural Networks*, **1**, 507–511.

GAREY, M.R. and JOHNSON, D.S. (1979) *Computers and Intractability.* W.H. Freeman, New York.

GEE, A.H. (1993) *Problem Solving with Optimization Networks.* Ph.D. dissertation, Queen's College, Cambridge University, U.K.

GHAZIRI, H. (1991) Solving Routing Problems by a Self-Organizing Map. In: T. Kohonen, K. Makisara, O. Simula, J. Kangas, eds., *Artificial Neural Networks.* North-Holland, Amsterdam, 829–834.

GHAZIRI, H. (1996) Supervision in the Self-Organizing Feature Map: Application to the Vehicle Routing Problem. In: I.H. Osman and J.P. Kelly, eds., *Meta-Heuristics: Theory & Applications.* Kluwer, Boston, 651–660.

GOLDSTEIN, M. (1990) Self-Organizing Feature Maps for the Multiple Travel-

ing Salesmen Problem. *Proceedings of the International Neural Network Conference*, **1**, 258–261.

GUERRERO, F., SMITH, K.A. and LOZANO, S. (1999) Self-organizing neural approach for solving the quadratic assignment problem. *Proc. Int. Workshop on Soft Computing in Industry*, 13–18.

GUERRERO, F., LOZANO, S., SMITH, K.A. and EGUIA, I. (2000) Facility Location using Neural Networks. In: Suzuki, Y. Ovaska, T. Furuhashi, R. Roy, and Y. Dote, eds., *Soft Computing in Industrial Applications*. Springer-Verlag, London, 171–179.

GUERRERO, F., LOZANO, S., CANCA, D. and SMITH, K.A. (1998) Machine Grouping in Cellular Manufacturing: A Self-Organizing Neural Network. *Proceedings 4th Int. Conf. on Engineering Applications of Neural Networks*. Systems Engineering Association, Turku, Finland, 374–377.

GUERRERO, F., LOZANO, S., SMITH, K.A. and KWOK, T. (1999) Manufacturing cell formation using a new self-organizing neural network. *Proc. Int. Conf. on Computers and Industrial Engineering*, **1**, 668–672.

HASEGAWA, M., IKEGUCHI, T. and AIHARA, K. (1997a) Combination of chaotic neurodynamics with the 2-opt algorithm to solve traveling salesman problems. *Physical Review Letters*, **79**, 2344–2347.

HASEGAWA, M., IKEGUCHI, T., MATOZAKI, T. and AIHARA, K. (1997b) An analysis on additive effects of nonlinear dynamics for combinatorial optimization. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, **E80-A**, 206–213.

HASEGAWA, M., IKEGUCHI, T. and AIHARA, K. (1998) Solving quadratic assignment problems by chaotic neurodynamics. *Proceedings of the 1997 International Conference on Neural Information Processing and Intelligent Information Systems*, **1**, 182–185.

HAYAKAWA, Y., MARUMOTO, A. and SAWADA, Y. (1995) Effects of the chaotic noise on the performance of neural network model for optimization problems. *Physical Review E*, **51**, R2693–R2696.

HEGDE, S., SWEET, J. and LEVY, W. (1988) Determination of Parameters in a Hopfield/Tank Computational Network. *Proceedings IEEE International Conference on Neural Networks*, **2**, 291–298.

HINTON, G.E., SEJNOWSKI, T.J. and ACKLEY, D.H. (1984) Boltzmann machines: constraint satisfaction networks that learn. *Carnegie Mellon University Technical Report* CMU-CS-84–119.

HOPFIELD, J.J. (1982) Neural Networks and Physical Systems with Emergent Collective Computational Abilities. *Proceedings of the National Academy of Sciences*, **79**, 2554–2558.

HOPFIELD, J.J. (1984) Neurons with Graded Response have Collective Computational Properties like those of Two-State Neurons. *Proceedings of the National Academy of Sciences*, **81**, 3088–3092.

HOPFIELD, J.J. and TANK, D.W. (1985) Neural Computation of Decisions in Optimization Problems. *Biological Cybernetics*, **52**, 141–152.

HOOKER, J.N. (1995) Testing Heuristics: We Have it All Wrong. *Journal of Heuristics*, **1**, 33–42.

JOHNSON, D.S. (1990) Local Optimization and the Traveling Salesman Problem. Volume **443** of Lecture Notes in Computer Science. "*Automata, Languages and Programming*". Springer-Verlag, Berlin, 446–461.

KAMGAR-PARSI, B. and KAMGAR-PARSI, B. (1992) Dynamical Stability and Parameter Selection in Neural Optimization. *Proceedings IEEE International Conference on Neural Networks*, **4**, 566–571.

KIRKPATRICK S., GELATT, C.D. and VECCHI, M.P. (1983) Optimization by Simulated Annealing. *Science*, **220**, 671–680.

KOHONEN, T. (1982) Self-Organized Formation of Topologically Correct Feature Maps. *Biological Cybernetics*, **43**, 59–69.

KOHONEN, T. (1988) *Self-Organization and Associative Memory.* Springer, Berlin.

KWOK, T., SMITH, K. and WANG, L. (1998a) Solving combinatorial optimization problems by chaotic neural networks. In: C. Dagli et al., eds., *Intelligent Engineering Systems Through Artificial Neural Networks: Neural Networks, Fuzzy Logic, Evolutionary Programming, Data Mining, and Rough Sets*, **8**, 317–322, New York: ASME Press.

KWOK, T., SMITH, K. and WANG, L. (1998b) Incorporating chaos into the Hopfield neural network for combinatorial optimization. *World Multiconference on Systemics, Cybernetics and Informatics*, **1**, 659–665.

KWOK, T. and SMITH, K. (1999a) A unified framework for chaotic neural network approaches to combinatorial optimization. *IEEE Transactions on Neural Networks*, **10**, 978–981.

KWOK, T. and SMITH, K.A. (2001) A noisy self-organizing neural network with bifurcation dynamics for combinatorial optimization. *IEEE Transactions on Neural Networks* (submitted).

LAI, W.K. and COGHILL, G.G. (1992) Genetic Breeding of Control Parameters for the Hopfield/Tank Neural Net. *Proceedings International Joint Conference on Neural Networks*, **4**, 618–623.

LIN, S. and KERNIGHAN, B.W. (1973) An effective heuristic algorithm for the travelling salesman problem. *Operations Research*, **21**, 498–516.

LO, J.T-H. (1992) A new approach to global optimization and its applications to neural networks. *Proceedings IEEE International Joint Conference on Neural Networks*, **4**, 600–605.

LOOI, C.K. (1992) Neural Network Methods in Combinatorial Optimization. *Computers & Operations Research*, **19**, 191–208.

LOZANO, S., GUERRERO, F., EGUIA, I. and SMITH, K.A. (1998) Cell Formation using Two Neural Networks in Series. In: C. Dagli et al., eds., *Intelligent Engineering Systems Through Artificial Neural Networks*, **8**, ASME Press, 341–346.

MATSUYAMA, Y. (1991) Self-Organization via Competition, Cooperation and Categorization Applied to Extended Vehicle Routing Problems. *Proceed-*

ings of the Int. Joint Conf. on Neural Networks, **1**, 385–390.

MCCULLOCH, W.S. and PITTS, W. (1943) A Logical Calculus of Ideas Immanent in Nervous Activity. *Bulletin of Mathematical Biophysics*, **5**, 115–133.

MINSKY, M. and PAPERT, S. (1969) *Perceptrons*, MIT Press, Cambridge, MA.

NEMHAUSER, G.L. and WOLSEY, L.A. (1988) *Integer and Combinatorial Optimization*, John Wiley & Sons, Canada.

NOZAWA, H. (1992) A neural network model as a globally coupled map and applications based on chaos. *Chaos*, **2**, 377–386.

PETERSON, C. and SODERBERG, B. (1989) A New Method for Mapping Optimization Problems onto Neural Networks. *International Journal of Neural Systems*, **1**, 3–22.

PETERSON, C. and SODERBERG, B. (1993) Artificial Neural Networks. In: C.R. Reeves, ed., *Modern Heuristic Techniques for Combinatorial Optimization*, Chapter 5, Blackwell Scientific Publishers, Oxford, UK.

POTVIN, J.Y. (1993) The Traveling Salesman Problem: A Neural Network Perspective. *ORSA Journal on Computing*, **5**, 328–348.

POTVIN, J.Y. and ROBILLARD, C. (1995) Clustering for Vehicle Routing with a Competitive Neural Network Model. *Neurocomputing*, **8**, 125–139.

ROSENBLATT, F. (1958) The perceptron: A Probabilistic Model for Information Storage and Organization in the Brain. *Psychological Review*, **65**, 386–408.

RUMELHART, D.E. and MCCLELLAND, J.L. (1986) *Parallel Distributed Processing: Explorations in the Microstructure of Cognition.* MIT Press, Cambridge, MA.

SIMMEN, M.W. (1991) Parameter Sensitivity of the Elastic Net Approach to the Traveling Salesman Problem. *Neural Computation*, **3**, 363–374.

SMITH, K.A. (1995) Solving the Generalized Quadratic Assignment Problem using a Self-Organizing Process. *Proceedings of the IEEE Int. Conf. on Neural Networks*, **4**, 1876–1879.

SMITH, K.A. (1996) An argument for abandoning the Travelling Salesman Problem as a neural network benchmark. *IEEE Trans. on Neural Networks*, **7**, 1542–1544.

SMITH, K.A. (1999) Neural Networks for Combinatorial Optimization: A Review of More than a Decade of Research. *INFORMS Journal on Computing*, **11**, 15–34.

SMITH, K.A., ABRAMSON, D. and DUKE, D. (1999) Efficient Timetabling Formulations for Hopfield Neural Networks. In: C. Dagli et al., eds., *Smart Engineering System Design: Neural Networks, Fuzzy Logic, Evolutionary Programming, Data Mining, and Complex Systems*, **9**, ASME Press, 1027–1032.

SMITH, K.A. and GUPTA, J.N.D. (2001) Continuous function optimisation via gradient descent on a neural network approximation function. Volume 2084 of *Lecture Notes in Computer Science. "Connectionist Models of Neurons, Learning Processes, and Artificial Intelligence"*. Springer-Verlag, Berlin, 741–748.

SMITH, K.A., PALANISWAMI, M. and KRISHNAMOORTHY, M. (1996) A Hybrid Neural Approach to Combinatorial Optimization. *Computers & Operations Research*, **23**, 597–610.

SMITH, K., PALANISWAMI, M. and KRISHNAMOORTHY, M. (1998) Neural Techniques for Combinatorial Optimization with Applications. *IEEE Transactions on Neural Networks*, **9**, 1301–1318.

SZU, H. and HARTLEY, R. (1987) Fast Simulated Annealing. *Physics Letters A*, **122**, 157–162.

SZU, H. (1988) Fast TSP algorithm based on binary neuron output and analog input using zero-diagonal interconnect matrix and necessary and sufficient conditions of the permutation matrix. *Proceedings IEEE International Conference on Neural Networks*, **2**, 259–266.

TAKEFUJI, Y. and SZU, H. (1989) Design of parallel distributed Cauchy machines. *Proceedings IEEE International Joint Conference on Neural Networks*, **1**, 529–532.

TANK, D.W. and HOPFIELD, J.J. (1986) Simple neural optimization networks: An A/D converter, signal decision circuit and a linear programming circuit. *IEEE Transactions on Circuit Systems*, **33**, 533–541.

TOKUDA, I., NAGASHIMA, T. and AIHARA, K. (1997) Global bifurcation structure of chaotic neural networks and its application to Travelling Salesman Problems. *Neural Networks*, **10**, 1673–1690.

VAKHUTINSKY, A.I. and GOLDEN, B.L. (1994) Solving Vehicle Routing Problems using Elastic Nets. *Proceedings of the IEEE Int. Conf. On Neural Networks*, **7**, 4535–4540.

VAKHUTINSKY, A.I. and GOLDEN, B.L. (1995) A Hierarchical Strategy for Solving Travelling Salesman Problems using Elastic Nets. *Journal of Heuristics*, **1**, 67–76.

VAN DER BOUT, D.E. and MILLER, T.K. (1988) A Travelling Salesman Objective Function That Works. *Proceedings IEEE International Conference on Neural Networks*, **2**, 299–303.

VAN DER BOUT, D.E. and MILLER, T.K. (1989) Improving the Performance of the Hopfield-Tank Neural Network through Normalization and Annealing. *Biological Cybernetics*, **62**, 129–139.

VAN DER BOUT, D.E. and MILLER, T.K. (1990) Graph partitioning using annealed neural networks. *IEEE Transactions on Neural Networks*, **1**, 192–203.

WANG, L. and SMITH, K.A. (1998) On chaotic simulated annealing. *IEEE Transactions on Neural Networks*, **9**, 716–718.

WILLSHAW, D.J. and VON DER MALSBURG, C. (1979) A Marker Induction Mechanism for the Establishment of Ordered Neural Mappings: Its Application to the Retinotectal Problem. *Philosophical Transactions of the Royal Society*, **Series B 287**, 203–243.

WILSON, G.V. and PAWLEY, G.S. (1988) On the stability of the TSP algorithm of Hopfield and Tank. *Biological Cybernetics*, **58**, 63–70.

WOLFE, W.J. (1999) A fuzzy Hopfield-Tank traveling salesman problem model. *INFORMS Journal on Computing*, **11**, 329–44.

YAMADA, T., AIHARA, K. and KOTANI, M. (1993) Chaotic neural networks and the Travelling Salesman Problem. *Proceedings of the International Joint Conference on Neural Networks*, **2**, 1549–1552.

ZHOU, D., YASUDA, K. and YOKOYAMA, R. (1997) A method to combine chaos and neural network based on the fixed point theory. *Transactions of the Institute of Electrical Engineers of Japan*, **117-C**, 599–608.