McMASTER
UNIVERSITY

MICHAEL G. DeGROOTE
SCHOOL OF BUSINESS

RESEARCH AND WORKING PAPER SERIES

SELF-ORGANIZING FEATURE MAPS: THE TRAVELING SALESMAN PROBLEM AND OTHER APPLICATIONS

By

Randy Bassett
and
Norman P. Archer

School of Business
McMaster University
Hamilton, Ontario
Canada

Working Paper # 386

October, 1993

ER UNIVERSITY
Street West
)ntario, Canada L8S 4M4
416) 525-9140

SELF-ORGANIZING FEATURE MAPS: THE TRAVELING SALESMAN PROBLEM AND OTHER APPLICATIONS

By

Randy Bassett
and
Norman P. Archer

School of Business
McMaster University
Hamilton, Ontario
Canada

Working Paper # 386

October, 1993

Self-Organizing Feature Maps:

The Traveling Salesman Problem and Other Applications

By

Randy Bassett

and

Norman P. Archer

Faculty of Business
McMaster University
Hamilton, Ontario
L8S 4M4

Self-Organizing Feature Maps:

The Traveling Salesman Problem and Other Applications

Abstract

Self-organizing feature maps (SOFM) have received much attention recently. SOFMs are basically a variant of neural network models which use unsupervised learning to acquire and organize their internal structure. Many mathematical features of the model have been discovered. In addition, many applications have been developed. This article is reviews the basic SOFM model as proposed by Kohonen. Next, using the traveling salesmen problem (TSP) as a benchmark, a few variants of the SOFM proposed to solve the TSP are compared to a variety of other algorithms such as the Hopfield/Tank network and simulated annealing. Brief descriptions of all algorithms are included. Lastly, a number of applications of the SOFM are discussed, such as speech and semantic recognition. The objective of this paper is to offer the reader some insight into the SOFM as well as some guidance as to further research.

Table of Contents

1) Introduction

Artificial neural networks (ANN's) have received much attention recently. This is largely due to the possibility of developing novel solution techniques for difficult problems with the hope of improving computational speed and performance. The term "neural networks" implies an association with biological phenomena, but for many ANN paradigms this association is quite superficial. The main feature of ANN's that mimic biological systems is the parallel processing of data. Common computers process data in a serial fashion. The significance of this difference can be easily demonstrated. The most advanced computers known today process an element of data in the nanosecond (10^{-9}) range whereas the brain, when prompted by external stimuli, has a cycle time in the millisecond (10^{-3}) range (Simpson, 1990). Although the brain processes information an estimated 6 orders of magnitude slower, it is superior at processing human information problems. This is due entirely to the large number of neurons in the brain processing data in a parallel manner.

As mentioned, numerous processing units (neurons) acting in a parallel fashion is the extent of the association between many ANN paradigms and biological phenomena. There is one system which has a stronger basis in biology and is useful in many non-biological applications. This is the self-organizing feature maps (SOFM's) first introduced by Teuvo Kohonen (1981). Biologically, this model has been widely accepted to explain the way in which the brain analyzes and processes sensory signals. For instance, there exists a *tonotopic map* in the auditory cortex of the brain. This map of perceived acoustic frequencies is perfectly ordered, almost logrithmically, with respect to frequency (Kohonen, 1989). These mappings are not genetically predetermined; they self-organize over the early stages of development of the nervous system.

As interesting as this model is as an explanation of some biological systems, it has been used as a computational tool in many areas such as medicine, engineering and business. Mathematical properties of this model are still being discovered and further discoveries should enhance its usefulness as a tool.

This paper has been written to highlight the properties and applications of the SOFM. The next section will discuss the basic model as proposed by Kohonen. The following section will discuss other algorithms proposed to solve the TSP. Following that will be a discussion of a number of variants of the SOFM model used to solve the Traveling Salesman Problem (TSP) and compare these systems to those previously discussed. The TSP was selected as a bench-mark for a variety of reasons. First, the amount of work done to discover improved solution mechanisms for this problem made it fairly easy to research a wide variety of algorithms to compare to the SOFM model. More importantly though, although the problem is easy to state, it is computationally difficult, yet it is broadly applicable to many engineering and business problems. The last major section of this paper will discuss applications of the SOFM. In this section examples will be discussed of how maps of higher than one dimension are used to help solve problems in robotics, pattern recognition and data compression. As well, an example of how SOFM's can discover semantic relationships in sentences will be discussed.

2) The Model

A very complete yet concise definition of the objective of SOFM's is to compress information by forming reduced representations of the most relevant facts, without loss of knowledge of their interrelationships (Kohonen, 1989). The SOFM is an unsupervised learning

model that is represented by a two-layer feed-forward topology. Essentially, analog vectors are presented to the input layer. After learning is completed, the weights will be clustered into vector centers that sample the input space. The point density function of these vector centers tends to approximate the probability function of the input vectors. Also, the weights will be organized such that nodes that are topologically adjacent will be sensitive to physically similar input (Lippmann, 1987). Qualitatively, the algorithm is as follows:

1) Initialize weights from N input vectors to M output vectors to small random values. Set the initial radius of the neighborhood to an arbitrary value. (The concept of the neighborhood will be discussed.)
2) Present input vectors.
3) Using a pre-set criteria, determine the node that best matches the input vector. This node is the so-called "winner". This is the competitive step in the algorithm.
4) Update (adapt) the weights of the winning node and those of the nodes within the winner's neighborhood. The radius of the neighborhood starts off wide and is systematically decreased as time (t) $\rightarrow \infty$ ("Time" in these simulation examples refers to successive steps or iterations of a process. Therefore, t is an integer value and not a continuous variable) .
5) Repeat steps 2 to 4.

To further discuss the model, the following notation will be utilized:

Let $x = [\xi_1, \xi_2, ..., \xi_n \mid \xi_i \in R \; \forall \, i]^T$

represent input vectors of scalar signals.

Let $\mu_{ij} \in R$

represent the weight from input node j to output node i.

Let $m_i = [\mu_{i1}, \mu_{i2}, \ldots, \mu_{in}]^T$

represent the *image* vector of output node i.

As mentioned above, values for all μ_{ij} are initialized to small random values. To determine the winning output node, denoted m_c, two methods are commonly used. The first involves the inner product $x^T m_i$ where

$$x^T m_i = \sum_{j=1}^{n} \mu_{ij} \xi_j .$$

$$x^T m_c = \max_i \{x^T m_i\} \quad \forall\, i = 1,2,\ldots,n$$

The second method is simply the normalized distances between x and m_i. The Euclidean norm is often used for simplicity. This second method may be more appropriate for certain applications such as natural signal patterns relating to metric vector spaces (Kohonen, 1990). Quite simply, the following criteria signifies the winning node, m_c:

$$\|x - m_c\| = \min_i \|x - m_i\| \quad \forall\, i = 1,2,\ldots,n.$$

Further possible modifications of the above two methods involve normalizing the weight vectors to constant length.

An interesting extension of the second method above is the introduction of a weighted Euclidean distance in the competitive step (Kangas et al., 1990). The square of the distance is defined as:

$$d_\omega^2[x(t), m_i(t)] = \sum_{j=1}^{N} \omega_{ij}^2 [\xi_j(t) - \mu_{ij}(t)]^2$$

where ω_{ij} is the weight of the jth component of the input vector x associated with cell i. These weights are estimated and adjusted throughout the learning stages. This method is useful when the variances of the components of $x = x(t)$ are significantly different. In this situation, if the

differences of the variances are not taken into consideration in the norm calculation an oblique orientation of the map may occur in which convergence to a sub-optimal vector quantization may result. A geometric interpretation of the resultant weighting of ω_{ij} is that the equidistant surface around cell i becomes elliptical. By setting $d_{\omega}[x(t),m_i(t)] = const.$, an N-dimensional ellipsoid results whose volume is:

$$V_i = C \prod_{j=1}^{N} \frac{1}{\omega_{ij}}$$

where C is a constant.

A further modification of the above process introduces a "conscience" into the system. The modification suggested is to stipulate

$$\prod_{j=1}^{N} \frac{1}{\omega_{ij}} = const., \quad \forall i.$$

The idea of a conscience is to prevent a node from winning too much. This concept was used by Burke and Damany (1992) in their development of a "guilty net". These modifications are discussed here to demonstrate some of the ways in which this part of the algorithm can be modified.

Figure 1 depicts how the system is configured. Each input pattern is presented to each of the vectors in the image vector layer (labeled "Kohonen layer" in the diagram). Through mechanisms previously discussed, a winning neurode is determined. Once the winning node (cell C) has been chosen, the weight parameter of the winner and those of the nodes within a defined neighborhood around the winner, N_c, are updated. This is the process which causes the clustering of related output neurons.

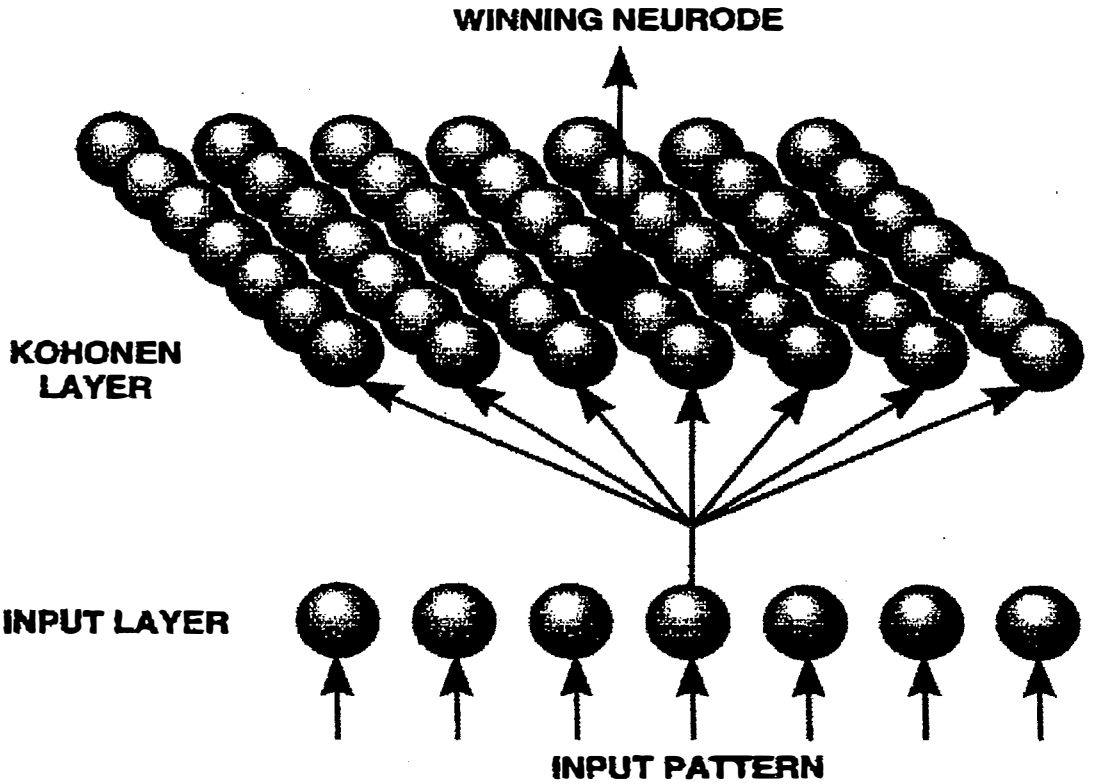

Fig. 1. Kohonen network (from Caudill, 1993)

Let N_c represent the defined neighborhood around the winning neurode. Figure 2 reflects the concept of a neighborhood. Although a rectangular neighborhood is shown, other shapes can be utilized such as hexagons. It is important to start with a fairly wide radius for N_c. If the initial neighborhood is too small various kinds of mosaic-like portions of the map are seen, between which the ordering direction changes discontinuously. The result is that the map will not be ordered globally (Kohonen, 1990). The initial radius of N_c can be more than half of the diameter of the network.

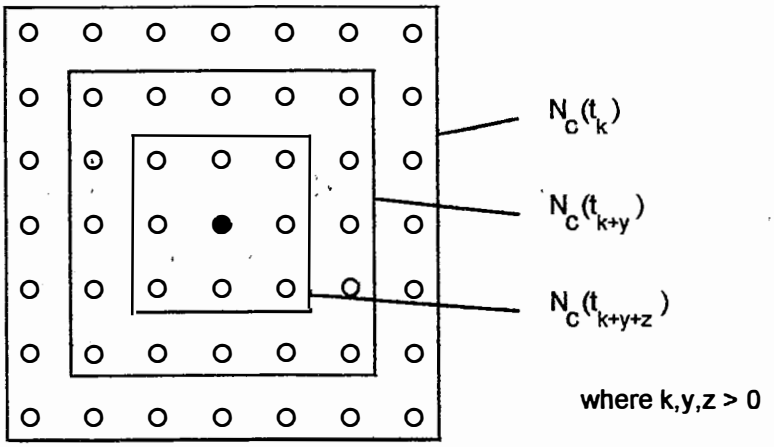

Fig. 2. Example of a topological neighborhood

This idea of a neighborhood allows updating of the m_i as blocks, concentrated around the winner. This process is used to obtain spatial ordering of the reference (or image) vectors. Owing to the frequent overlap of these neighborhoods, the values of the m_i tend to be smoothed. Also, they tend to become ordered, especially at the borders of the network (Kohonen, 1990).

There are many variants of the updating or learning laws in the literature. One such law, used in conjunction with the Euclidean norm type of matching rule is:

$$m_i(t+1) = m_i(t) + \alpha(t)[x(t) - m_i(t)]$$

$$\forall\, i \in N_c(t)$$

$$m_i(t+1) = m_i(t)$$

$$\forall\, i \notin N_c(t).$$

In this formulation, $\alpha = \alpha(t)$, termed the adaptation parameter, can take on many forms. It could be a simple scalar parameter that decreases monotonically over time ($0 < \alpha < 1$). An alternative notation is to introduce a "kernel" function, h_{ci}:

$$m_i(t+1) = m_i(t) + h_{ci}(t)[x(t) - m_i(t)]$$

This notation is equivalent to the previous example if $h_{ci}(t) = \alpha(t)$ within N_c, and $h_{ci}(t) = 0$ outside N_c. Alternatively, a more general form of h_{ci} may be used. Let r_c and r_i represent the coordinates of cells C and i respectively. A common form for h_{ci} is the Gaussian

$$h_{ci} = h_0(t) \exp(-\frac{\|r_i - r_c\|^2}{2\sigma_E^2(t)})$$

with $h_0(t)$ and $\sigma_E(t)$ as suitable decreasing functions of time (Kohonen, 1990). The parameter σ_E represents the width of the Gaussian function. In this example, it determines the length scale on which the input vectors cause corrections to the map (Ritter et al. 1992). Normally, $\sigma_E(0)$ is rather large. Gradually $\sigma_E(t)$ decreases over time which effectively increase the "selectivity" of the individual neurons over the learning phase.

Another learning law worth noting is the following example in which the image vectors are normalized at each step:

$$m_i(t+1) = \frac{m_i(t) + \alpha'(t)x(t)}{\|m_i(t) + \alpha'(t)x(t)\|}, \quad \forall i \in N_c(t)$$

$$m_i(t+1) = m_i(t), \quad \forall i \notin N_c(t)$$

where $\alpha' = \alpha'(t)$ is a monotonically decreasing scalar function of time, but now $0 < \alpha' < \infty$ and $N_c(t)$ is a similar neighborhood around the "winner" as in the previous example (Kangas et al., 1990). As an example, $\alpha' = \alpha_0 / t$ could be used where α_0 is large in the range of 10 to 100.

In order to demonstrate the above-mentioned processes, consider the following example taken from Kohonen (1990). The input vectors are two-dimensional for visual display purposes. Also, the probability density function of the input vectors are arbitrarily selected to be uniform over the area delimited by the triangular (figure 3) or rectangular (figure 4) border. The density is zero outside the borders. A set of input vectors $x(t)$ were independently and randomly selected from from this density function. This set was used to cause changes in the weight vectors m_i.

In the example depicted in figure 3 the weight vectors appear in the same coordinate system in which the $x(t)$ are represented. The processing unit array is two-dimensional which coincides with the dimension of the distribution. More interesting (and more common) is the situation in which the dimensions of the distribution and that of the array differ. In figure 4 the distribution is two-dimensional but the array is one-dimensional. Quite simply, a one-dimensional output array is linear chain of cells. The weight vectors of these linear arrays tend to approximate to higher-dimensional distributions by Peano curves (Kohonen, 1990).

A couple of general points should be made about the SOFM paradigm. The model thus far discussed does not classify its output as do paradigms such as backpropagation. Rather, the

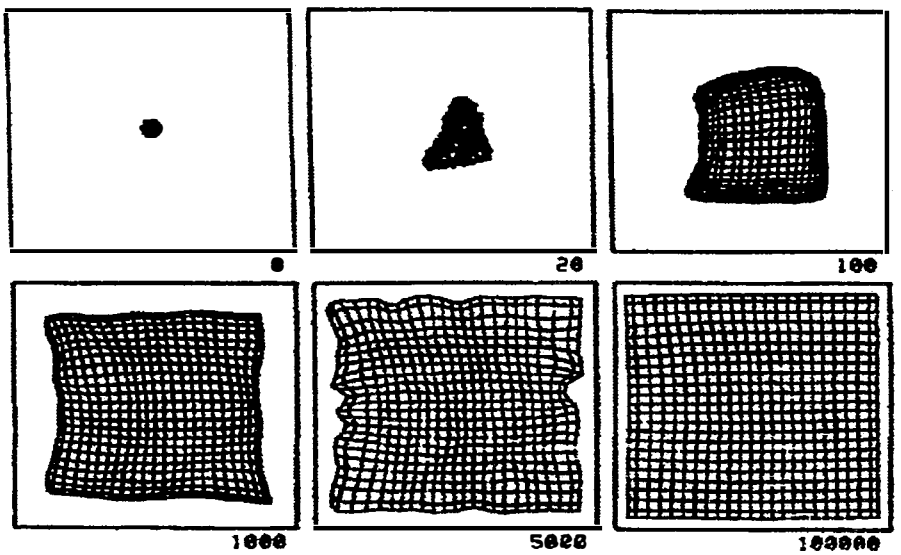

Fig. 3. Two-dimensional array of weight vectors during the ordering process. (From Kohonen (1990)).

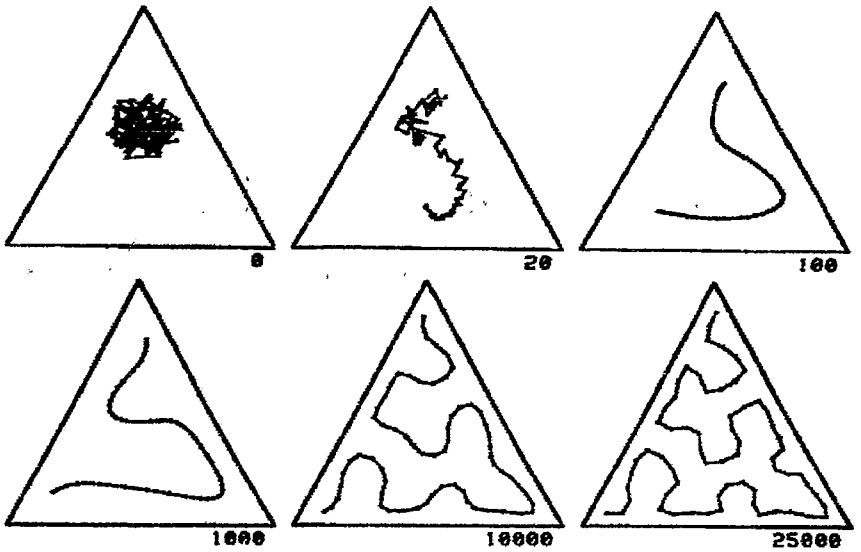

Fig. 4. One-dimensional array of weight vectors during the ordering process. (From Kohonen (1990)).

output nodes act more of a representative of a particular class of input cases than as any particular feature that belongs to that class (Hiotis, 1993). Subsequently, the basic SOFM system as thus far discussed should not be used by itself for pattern recognition or any other type of decision processes. Rather, it is possible to fine tune the model to significantly increase the recognition accuracy of the system (Kangas et al., 1990). One such method of fine tuning is called Learning Vector Quantization (LVQ).

The LVQ model can be highlighted with a brief discussion of Type One Learning Vector Quantization (LVQ1) (Kohonen, 1990). In order to classify the input into a finite number of classes it is essential to assign several "code book" vectors, m_i, to each class. It is important to note that the identity of these vectors *within* the class is not crucial, rather, only the decisions at the class borders are significant. By defining effective values for the code book vectors the aim is to directly define near-optimal decision-borders between the classes. The first step is to

initialize the values of m_i using an algorithm such as the SOFM. The next step is to determine the labels of the code book vectors by presenting a number of vectors of known classification. The accuracy of this classification can be improved if the following laws for updating m_i are used:

$$m_c(t+1) = m_c(t) + \alpha(t)[x(t) - m_c(t)]$$

if x is classified correctly,

$$m_c(t+1) = m_c(t) - \alpha(t)[x(t) - m_c(t)]$$

if the classification of x is incorrect,

$$m_c(t+1) = m_c(t) \quad \text{for } i \neq c$$

where c represents the class being "tested". The gain function, $\alpha(t)$, is a monotonically decreasing function of time, similar to that used in SOFM. The key difference is that in the LVQ1 algorithm, $\alpha(0)$ is set significantly lower than in the SOFM because the LVQ1 is a fine-tuning method. The central idea behind the above learning law is to pull code book vectors away from the decision surfaces to increase the accuracy of the demarcation of the borders (Kohonen, 1990). Each m_i has an area of attraction. The sum of these within a particular class defines the area of that class. Therefore, using multiple reference vectors for each category allows the system to define asymmetric or oddly shaped class boundaries. This gives this type of system greater power in solving difficult problems (Caudill, 1993).

As demonstrated by the previous discussion, the SOFM has a number of parameters and steps that can be modified to arrive at a variety of different specific models. The derivation of some particular models will be discussed in the following sections regarding proposed mechanisms to solving the TSP.

3) The Traveling Salesman Problem

i) An Overview

Simply stated, the TSP consists of finding the shortest possible tour through a set of n cities, visiting each and every city exactly once. The TSP is a classical example of a combinatorial problem which is NP-complete. NP-complete problems are those in which no polynomial bound algorithms have been found. That is, the number of operations required to arrive at an optimal solution increases faster than any power of N where N is the size of the problem (i.e. the number of cities in the TSP). Since the tour is a complete loop of all cities, there are $\frac{1}{2}(N-1)!$ possible tours. Even for a relatively small value of $N = 30$, the total number of tours exceeds 4.4×10^{30}. Even using a Cray-XMP supercomputer, the processing time to complete an "exhaustive search" would exceed the age of the universe (Ritter et al., 1992). Since many problem types, such as task scheduling in operations management, fall into this class of NP-complete problems, there is much need and interest in developing algorithms which give near-optimal solutions in polynomial time to these types of problems.

ii) A Review of Proposed Solution Mechanisms

The literature is rich with various algorithms put forth to solve the TSP. Lin and Kernighan (1973) proposed a procedure that is based on a general approach to heuristics. It has been shown to be very efficient and run times grow about as N^2.

More recently, Kirkpatrick et al. (1983) used an optimization technique called simulated annealing to tackle the TSP. This work triggered a breakthrough to applications on NP-problems. This method involves a random generation of "moves" or rearrangements of elements (i.e. cities) in the system to arrive at a set of possible solutions. The mechanism of generating such

moves was borrowed from Lin and Kernighan (1973). If E is a cost or an energy function, classical gradient descent algorithms only accept moves in which $\Delta E < 0$. The problem is that there is no way to escape local minima. On the other hand, simulated annealing also accepts moves in which $\Delta E \geq 0$ with the Boltzman probability that the move is accepted being $P(\Delta E) = \exp(-\Delta E / k_b T)$. The parameter T is called the annealing temperature. A vital component of the algorithm is the annealing schedule of the value of T. If the schedule decreases T too rapidly the system may become trapped in a local minima. Therefore, it important that T be decreased slowly so that E can converge to a state of minimum energy. Although this method has theoretically been shown to converge to globally optimal solutions (Aarts and Korst, 1989), it becomes only an approximate algorithm in practical implementations, for the convergence to globally optimal solutions only holds asymptotically (Looi, 1992). Kirkpatrick et al. (1983) used this method to anneal into optimal solutions with confidence levels greater then .95 for N up to 6,000 sites.

The first connectionist approach to the TSP arose with the work of Hopfield and Tank (1985). Their network was a non-adaptive system seeking a minima of a particular energy function. The output layer requires N^2 nodes, for it represents an $N \times N$ matrix of N cities vs. N positions in a given tour. This layer is entirely interconnected resulting in N^4 interconnections. The weights between each pair of neurons are predetermined and fixed. As applied to the TSP, the energy function in the Hopfield and Tank model takes the following form:

Let $V_{x,i} = 1$ if city x is visited in the ith position, $x,i \in [1,2,...,N]$

$V_{x,i} = 0$ otherwise.

The energy function is defined as:

$$E = \frac{A}{2} \sum_{x} \sum_{i} \sum_{j \neq i} V_{xi} V_{xj}$$

$$+\frac{B}{2}\sum_{i}\sum_{x}\sum_{y\neq x}V_{xi}V_{yi}$$

$$+\frac{C}{2}[(\sum_{x}\sum_{i}V_{xi})-N]^2$$

$$+\frac{D}{2}\sum_{x}\sum_{y\neq x}\sum_{i}d_{xy}V_{xi}(V_{y,i+1}+V_{y,i-1})$$

where d_{xy} is the Euclidean distance between cities x and y. The first term is zero if and only if each city is visited at most once. The second term is zero if and only if each position on the tour has no more than one city. The third term is zero if and only if there are N entries in the N^2 solution matrix. Therefore, the first three terms are zero if and only if a feasible tour is represented in the solution matrix. The last term of E refers to the tour distance for which, of course, a minimum is being sought. This energy function is analogous to a penalty (Lagrange) function (Burke and Damany, 1992). Using E as defined here along with the fixed weights of the interconnections, the network is allowed to dynamically alter its state until it converges to a minimum for E.

Numerous shortcomings of the Hopfield and Tank approach have been cited. First, the computational efforts of the model are huge. This optimization problem is not solved in the problem space of cardinality $N!$, but rather in the representation space of cardinality 2^{N^2} (Looi, 1992). Also, some researchers were unable to replicate Hopfield and Tank's results (Wilson and Pawley, 1988). Furthermore, the result obtained by Hopfield and Tank (1985) for only a 30 city problem was 19% longer than the solution found by the Lin and Kernighan method (Xu and Tsai, 1991).

Another connectionist approach to the TSP found in the literature involves Markovian neural networks (Kovacic, 1991). This system consists of n neurons. As Markov chains consist

sets of states and the associated probabilities of transition to successor states, let $S = \{s \mid s = (x_1, x_2, ..., x_n)\}$ be a finite set of states. Each state is described with n parameters and every neuron in the Markovian neural network represents one element (x_i) of state vector $s \in S$. Also, let f: $S \rightarrow \Re$ be an evaluation function over the states. The objective is to find the best s with respect to f , that is, find such s such that $\forall\, s_i \in S : f(s) \leq f(s_i)$. Let s_i be the current state of the network. Kovacic (1991) defined the likelihood ratio of transition from the current state s_i to s_j as

$$r_{ij}^{(T)} = \frac{P^{(T)}(s_i \rightarrow s_j)}{P^{(T)}(s_i \rightarrow s_i)} = \exp\left(\frac{f(s_i) - f(s_j)}{T}\right)$$

where $s_i \rightarrow s_j$ denotes the transition from s_i to s_j. The parameter $T > 0$ is called the temperature and is decreased in every iteration in a manner similar to that used in simulated annealing. In his article, Kovacic presents a proof of the equivalence of Markovian neural networks and Markov chains.

As applied to the TSP, Kovacic reported that the complexity of the Markovian neural network simulation grew proportionally to N^2. This architecture outperformed other methods tested including simulated annealing. The main reason is that the Markovian system takes into account all one-step transitions while computing transition probabilities, whereas other methods such as simulated annealing consider only one transition at a time.

The last approach to the TSP to be discussed in this section involves genetic algorithms. Genetic algorithms (and more broadly genetic programming) involve searching large portions of a solution space for solutions (offspring) which are more "fit" than their "parent" solution states. In fact, this system is highly parallel in that the search through a solution space is done simultaneously by an entire population of "individuals". This is akin to natural phenomena in that nature generally adapts populations of individuals and not just individuals themselves (Banzhaf,

1990). Generally, recognized mechanisms such as genetic recombination, mutation, inversions and crossovers are programmed into specific search algorithms. For instance, Fogel (1988) used an approach in which offspring were generated through random mutation of each parent by choosing a city in a parent's tour list and replacing it in a different, randomly chosen position. Also, this particular approach incorporated the probabilistic aspect of survival witnessed in nature. That is, individuals possessing some kind of advantage are not guaranteed survival, rather they only have an increased probability of survival. This search mechanism was shown to be very successful. Using a formula for calculating the expected optimal length derived from Bonomi and Lutton (1984) and a normal approximation to the error distribution of tour lengths, the mean best tour length of ten trials of a 100 city TSP was superior to 99.99999999999% of the possible tours. The impressive factor is that only 8.58×10^{-151} of the total number of tours were evaluated. Figure 5 shows the graph of the best mean tour length vs. number of evaluated offspring for this problem. In a 100 city TSP there are approximately 4.666×10^{155} different tours. The graph in figure 5 shows the asymptotic approach to the best mean tour length with relatively few offspring evaluated thus reflecting the search efficiency of genetic algorithms. Banzhaf (1990) also reported success using evolutionary optimization for the TSP. These findings suggest that genetic algorithms may be efficient searching tools which, when combined with neural network classifiers such as LVQ1 and backpropagation may prove to be very helpful (Fogel et al., 1990; Brill et al., 1992).

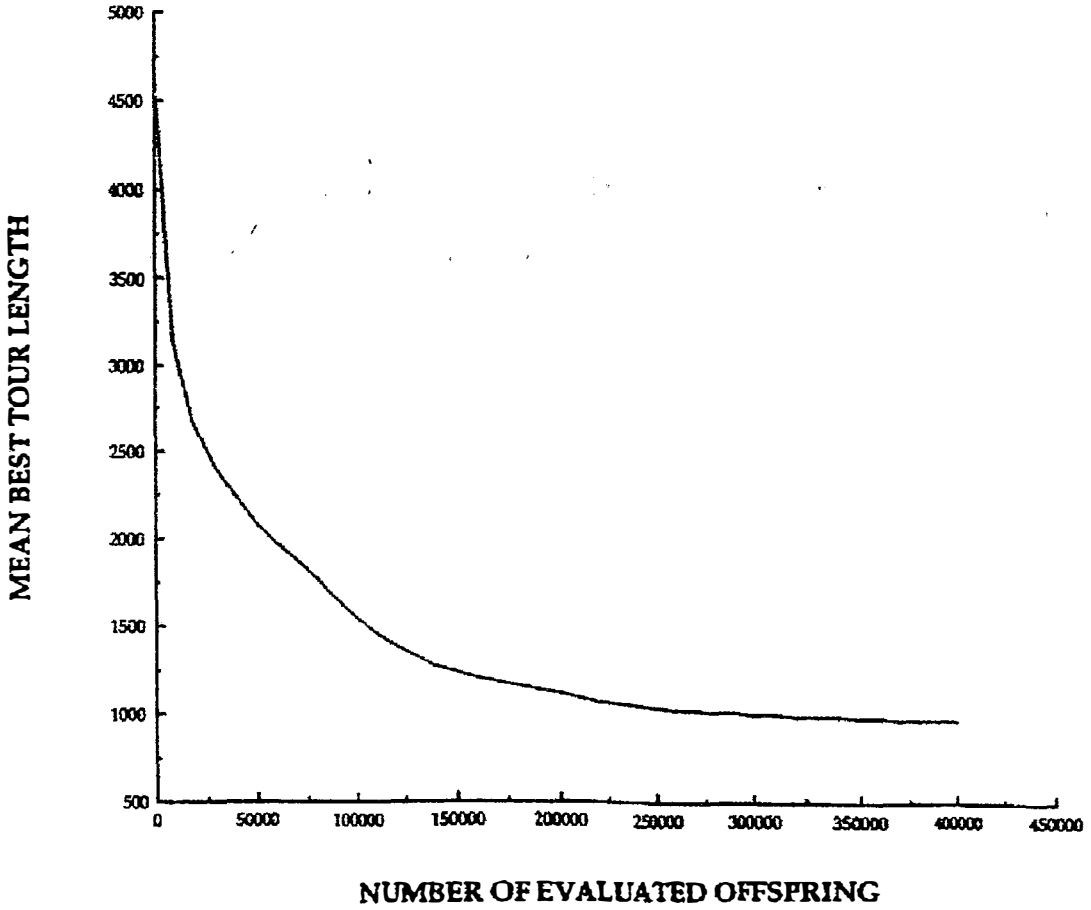

Fig. 5. Evolutionary optimization. Average of ten trials of a 100 city TSP. Taken from Fogel (1988).

In this section four different solution methods for the TSP were discussed. This was obviously not an exhaustive listing of methods proposed in the literature. The discussion here was put forth to give a brief overview of each approach, thereby offering some understanding of the respective models. The sample of algorithms chosen for this report were carefully selected with a couple of objectives in mind. First, the sample reflects a wide variety of algorithms. For example, the Hopfield network was chosen because of its historic significance, being the first neural network application to NP-complete problems such as the TSP. Also, genetic algorithms

were selected because they are powerful searching tools that are very different from neural network algorithms. Second, this sample includes algorithms commonly used as benchmarks against which newly-proposed systems are tested. For instance, simulated annealing is frequently used as a benchmark against other TSP algorithms due to its ability to discover relatively low tour lengths. The Markov neural network was selected because it was reported that it out-performed simulated annealing. This fact indicates that the Markov neural network may be a better benchmark than simulated annealing. Thus, this sample of algorithms satisfied the objective of covering a wide variety of algorithms while also discussing methods frequently referenced as benchmarks.

iii) Solutions Using Self-Organizing Feature Maps

The first self-organizing model to be discussed is the elastic net approach proposed by Durbin and Willshaw (1987). The algorithm starts with k points lying on an imaginary "elastic band" where $k > N$ (N is the number of cities). The points in this band "move" until eventually all cities in the solution space are "caught" by a node. This movement is influenced by two types of forces. The first moves a node closer to cities to which it is nearest and the second pulls it towards the neighbors on the path, thereby acting to minimize tour length. With this approach, each city becomes associated with a particular section of the path. The tightness of the radius in which this association is strengthened is dependent on a parameter K. Throughout the simulation this parameter is lowered in a manner loosely equivalent to the "temperature" in simulated annealing. Denoting the coordinates of city i by the vector x_i and those of a typical point j on the elastic net as y_j, then the rule for change Δy_j at each iteration is defined as:

$$\Delta y_j = \alpha \sum_i w_{ij}(x_i - y_j) + \beta K(y_{j+1} - 2y_j + y_{j-1}) \qquad (1)$$

where α and β are constants defining the relative forces of two forces described above and w_{ij} denotes the influence of city i on the elastic path point j. The latter parameter is normalized so that the total influence of each city is equal:

$$w_{ij} = \frac{\phi(|x_i - y_j|, K)}{\sum_k \phi(|x_i - y_k|, K)} \qquad (2)$$

$\phi(d,K)$ is the Gaussian function $\exp(-d^2 / 2K^2)$. The energy function is defined as:

$$E = \alpha K \sum_i \ln \sum_j \phi(|x_i - y_j|) + \beta \sum_j |y_{j+1} - y_j|^2 \qquad (3)$$

which has the property that

$$\Delta y_j = -K \frac{\delta E}{\delta y_j} \qquad (4)$$

This means that for any change in y_j as in equation (1), a reduction in the value of E results. Since E is bounded below, local minima for E will eventually be reached (Durbin and Willshaw, 1987).

Applied to a 30 city TSP, the elastic net method was superior to Hopfield and Tank's method (the latter generated tours which were on average 19% longer than those generated by the former). The results obtained for a 50 city simulation were only 1.5 $\pm$0.7 % longer than the results obtained from simulated annealing in approximately the same computing time. As this algorithm is fundamentally geometric, it can be extended to more general TSP problems in Euclidean space, but not to cases where arbitrary matrices of distances are given.

Another variant of the SOFM model used to solve the TSP was proposed by Angeniol et al. (1988). This approach uses elastic net concepts but is implemented as a SOFM. Using the

notation in the work by Angeniol et al. (1988), the cities are numbered 1 to *M*, the nodes of the output map are numbered 1 to *N*, each city *i* is denoted by its two coordinates: (x_1^i, x_2^i) and each node *j* is identified by the two coordinates (c_1^j, c_2^j). Also, each node is related to its two immediate neighbors in the ring. Each complete iteration of the system takes *M* steps, picking every city in the application once in a fixed order. This order is picked randomly prior to running the simulation and different solutions result from different initial orders.

The simulation begins with only one node located in the plane at the point (0, 0). There is a node creation (deletion) mechanism in the system which increases (decreases) the number of nodes in the network. When a city is "surveyed" by the network, a competition based on Euclidean distance is conducted to see which node of the net is closest to the city. The winning node and its neighbors are moved towards the city by a distance determined by a function

$$f(G,n) = \frac{1}{\sqrt{2}} \exp\left(\frac{n^2}{G^2}\right)$$

where *n* is the distance along the ring between nodes j_c (the winning node) and arbitrary other nodes *j*; and *G* is a gain parameter. Each node is moved from its current coordinates (c_1^j, c_2^j) to a new position by the equation:

$$c_k^j \rightarrow c_k^j + f(G,n) \cdot (x_k^i - c_k^j).$$

By virtue of the Gaussian definition for the function *f*, as $G \rightarrow \infty$, all nodes move towards city *i* with the same strength $(\frac{1}{\sqrt{2}})$, whereas as $G \rightarrow 0$, only the winning node, j_c, moves towards city *i*.

Between each iteration the gain parameter is adjusted by:

$$G \rightarrow (1 - \alpha) \cdot G.$$

Therefore, α is the only parameter requiring tuning.

According to Angeniol et al. (1988), node creation and deletion are necessary for success. Node creation occurs when one node has won the competition for two different cities in one iteration. The new node is given the same coordinates as the winner. Both the present winning node and the newly created one are inhibited in the next competition. This enables the non-overlapping neighbors of these nodes to pull them apart. Node deletion occurs if a node has not won a competition for 3 complete surveys.

The results obtained when applied to the TSP were quite promising. Although the reported results of tour lengths were slightly higher than those calculated using other algorithms such as the elastic net method, the computational time was impressive. Lower values for α gave slightly better tour lengths, but the simulations took longer (due to the fact the gain parameter is decreased in smaller intervals between iterations). For instance, simulations were run on a 1,000 city TSP. It took 12 hours to reach a reasonable solution (length = 18,036) using $\alpha = 0.01$, whereas when $\alpha = 0.2$ was used, tour lengths ranging from 18,200 to 18,800 were found in only 20 minutes. (The authors did not mention what computer was used for their simulations).

There are some concerns with the work of Angeniol et al. (1988). For instance, they mentioned that the fixed order of cities used in the simulation gave different results for different initial orders. They did not elaborate on the extent of these differences. As promising as their results appear, obviously more work is necessary to elucidate the underlying mechanisms of their system.

The key result from their work is the demonstration of the scaling of the problem size with large numbers of cities. As shown in figure 6, for the 10,000 city problem, the net has to cycle for only 4 times the number of cities. It seems that the net is fully exploiting the greater

smoothness in the solution space for larger problem sets. Also, the number of cycles required for problem sizes between 1,000 and 10,000 cities is roughly constant at 40,000. Since the time taken to execute a single iteration increases linearly with problem size, this represents a linear increase in computing time with respect to problem size on a serial machine. Figure 7 reflects the typical solutions for a 1,000 and a 10,000 city TSP.

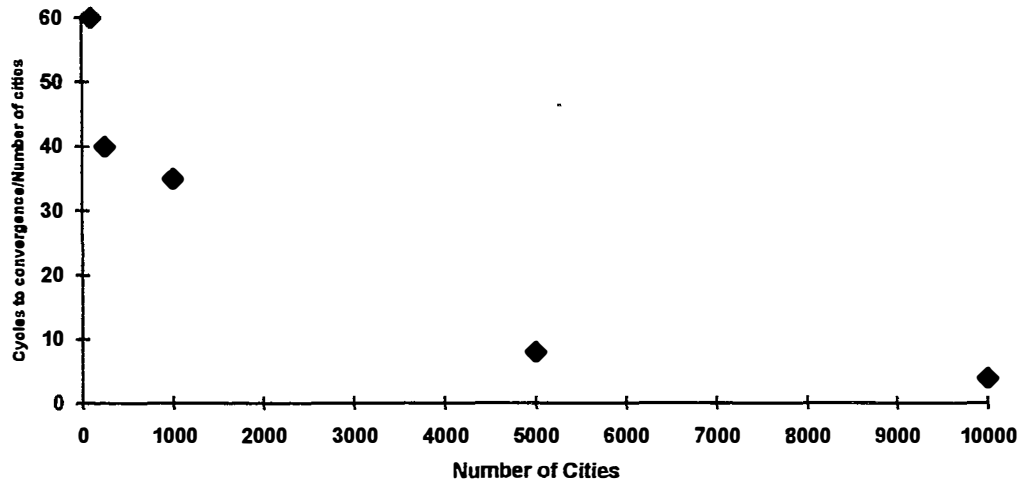

Fig. 6. Number of cycles necessary for convergence to 10% of the simulated annealing solution vs. problem size (Favata and Walker, 1991).

Favata and Walker (1991) also applied a variant of the SOFM to the TSP. Their model is exactly like that described in an earlier section this report (see middle of page 8) in which image vectors are normalized at each step. For each city, the input vector has three components: the first two being the coordinates of the city, the third being a normalization component computed so that all input vectors have the same Euclidean length and no two input vectors are collinear. The gain reduction (α) schedule was the same as that recommended by Kohonen (1989): in the first 10% of the iteration cycles the gain is reduced linearly to 10% of its starting value and over

the remaining 90% of the iterations it was linearly reduced to zero. The results of Favata and Walker's work show low sensitivity to initial values of the gain parameter. The criteria for convergence in their work is that the average path length over a sample of 50 solutions must be within 10% of the solution found using simulated annealing.

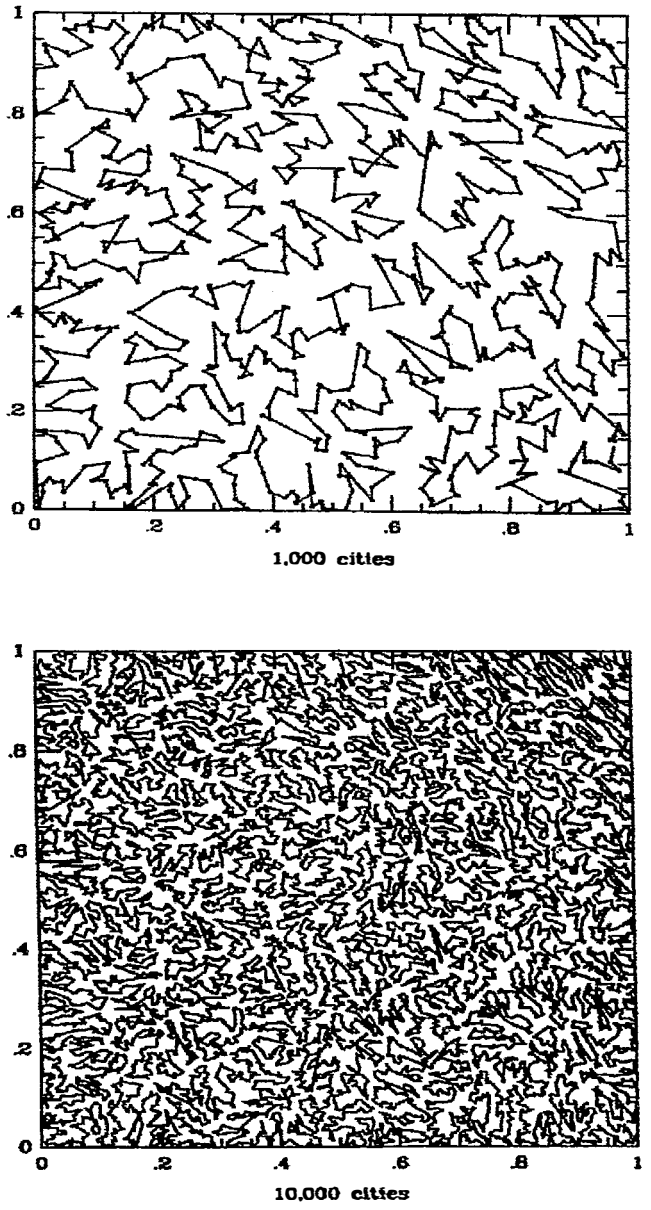

Fig. 7. Typical solution for the 1,000 (upper) and 10,000 (lower) city TSP. Taken from Favata and Walker (1991).

Favata and Walker also noted that the time taken to achieve a reasonable solution for a 10,000 city TSP with their method was 10 times faster than that obtained using simulated annealing. This therefore represents a trade-off between two factors: time and quality of solution. Depending on the application, one of these two usually opposing factors is dominant. Thus, this demonstrates the need to consider qualitative issues in selecting appropriate neural network architectures.

The final model to consider in this section is the so-called "guilty-net" proposed by Burke and Damany (1992). The output layer of this system has exactly N nodes, where N is the number of cities in the problem. Each of these nodes has exactly 2 neighbors, thereby resulting in a ring structure. The interesting aspect of this model is the introduction of a "conscious mechanism" in the competition step. For each node j a bias term is calculated:

$$\text{bias}(j) = \frac{\text{win}(j)}{\{1 + \sum_k \text{win}(k)\}}$$

where win(i) represents the number of wins for node j. A modified competition is held with the winner defined as:

$$\|x - m_c\| = \min_j \{\|x - m_j\| + K \cdot \text{bias}(j)\} \qquad \forall\, j = 1,2,\ldots,N$$

where K is a constant (set at 10 in the simulations in this work). Updating of weights is done in a similar fashion to the basic SOFM model discussed earlier in this paper.

The idea of a conscience mechanism was first introduced by DeSieno (1988) as a means to produce equiprobable weights in learning systems such as SOFM. The objective is to inhibit nodes that are winning "too often". The bias term in the above equations starts off low and increases throughout the simulation. Therefore, all nodes have a higher likelihood of winning early in the process. As the process continues, the increasing conscious term for various nodes

tends to cause a separation of nodes; one node per city. As compared to the work of Angeniol et al. (1988), the conscience mechanism acts in place of node creation and deletion.

The reported results of simulations run using the guilty net model shows that this model gave poorer results than runs using simulated annealing or elastic net approaches, but computationally fewer steps were involved.

This last model exemplifies the current trend in neural network research. Applications of neural networks are being advanced much faster than the theoretical frameworks upon which applications could be more deftly developed. For instance, in the guilty net model a parameter *K* was introduced and set to 10. Presumably there exists flexibility in the pre-set value for *K*, but a lack of a theoretic framework limits the widespread application of such a parameter to little more than empirical studies. As a further example, Burrascano (1991) used the LVQ1 variant of the SOFM in the learning phase of a probabilistic neural network (PNN). The PNN is a pattern classifier and the LVQ1 was used to simplify the network structure. Burrascano discusses the flexibility of choosing a number of nodes in one particular layer as a "trade-off between acceptable network complexity and desired classification accuracy". This demonstrates further the gap between application development and theoretical understanding. Although this approach to neural network research appears limiting, it is highly justified for at least two reasons. First, it gives theorists a wide base of "observations" and interpretations to draw on. Second, the usefulness of neural networks have met with some doubt in the recent past. Therefore, a library of demonstrated applications of these systems further justifies the expenditure of time and resources in studying these systems.

The four models of self-organizing maps discussed in the section were selected to reflect the variations in the SOFM model used to solve problems such as TSP. Favata and Walker's

method is very similar to the basic Kohonen model previously described and it is able to solve relatively large TSP problems (10,000 cities). The elastic net method was chosen because it was the first self-organizing approach to the TSP. This method is also quite effective and efficient in finding a good solution. The method introduced by Angeniol et al. is a derivative of the elastic net method with mechanisms for node creation and deletion included. Lastly, the guilty net method imposed a conscious mechanism onto the SOFM model. It should be apparent from this discussion that many variations of the SOFM are possible. This variability makes the SOFM model widely useful in many applications, some of which will be discussed in the following section.

4) Applications of the Self-Organizing Feature Maps

As previously mentioned, there are a number of applications of the SOFM in the literature. A few of these will be discussed including data compression, integration with expert systems, speech recognition and semantic recognition.

The first application area to address is statistical data compression. Gersho and Shoham (1984) introduced the idea of using vector quantization to compress speech data. The general idea is to find a set of disjoint regions $A_1, A_2, \ldots, A_{2^M}$ of $\Re^n$ such that

$$\bigcup_{k=1}^{2^M} A_k = \Re^n .$$

These regions are to be selected so that a data vector x that is chosen with respect to some fixed probability density function is equally likely to be in each region. If x is a data vector to be transmitted and A_k is the region containing x, then the index k of the region is transmitted instead of x. In order to have reliable reconstruction of the compressed data, the original input vectors have to be classified into the proper region with a sufficiently high confidence level. A variation of the SOFM called counterpropagation has been cited as an appropriate classifier (Hecht-Nielsen, 1988). The counterpropagation model couples Kohonen's SOFM with the outstar model of Grossberg (1982) to arrive at an effective and efficient system. (See Hecht-Nielsen (1988) for details of the counterpropagation model). An advantage of the counterpropagation model over other proposed systems is that the A_k regions usually turn out to be highly isotropic. Isotropic regions are better at classifying vectors. For instance, rectangles in high dimensions (n) are such that the distance from the center of the rectangle to the center of each of the $2(n-1)$ faces is quite small whereas the distance form the center to each of the 2^n vertices is very large. These regions are highly anisotropic and it is difficult to properly classify vectors lying in the

corners of such regions. A further advantage of counterpropagation for data compression is that these networks can be used in a hierarchical manner to achieve even more compression although some performance may be lost.

Another interesting application of the SOFM model involved expert systems. One of the most difficult issues in the design of expert systems is the problem of attribute selection for knowledge representation. Tirri (1991) has designed a system which uses SOFM to determine those properties of data that reflect meaningful statistical relationships in the expert system input space. The basic SOFM model is applied with each input instance defined as a vector $v = (v_1, v_2, ..., v_d)$ of a d-dimensional input space S. The training set T is a set of such vectors. After completion of the training process each cluster C_i in the output node "grid" is labeled with a meaningful attribute name. The labeling is done by finding an example set of vectors from the training set T such that the nodes in C_i are sensitive to these input instances. Better classification can result by using a more sophisticated back-end to the SOFM (e.g. LVQ methods).

The above approach was applied to two different systems: automatic inspection of circuit packs and intelligent monitoring of medical equipment (Tirri, 1991). Focusing on the latter, a significant reduction in the number of rules in an expert system involving respiratory and anesthesia monitoring can be achieved. In the revised model, detector predicates are used to perform the pattern recognition function of determining which "cluster" an input signal belongs to. With this approach, the rule base is free to describe only the necessary actions. The expected decrease in the number of rules in the rule base of the resulting "delivered" expert system is huge (~98%).

A further application of the SOFM involving expert systems concerned knowledge acquisition (Coleman and Watenpool, 1992). These authors discussed the issues of knowledge acquisition using a fault-identification expert system. Although this discussion involving expert systems and SOFM is quite promising, much more work has to be done to facilitate the integration of expert systems and neural networks.

Another application area of interest involves speech recognition. Owing to temporal and spectral variations in speech, unconstrained human voices are very difficult to recognize. Neural networks are not well suited to deal with temporal variations. One model proposed to deal with these variations is called the hidden Markov model (HMM) (Zhao and Rowden, 1992). It is known that a phoneme (smallest unit of distinguishable speech) has tremendous temporal and spectral variations. HMMs are very successful for modeling these variances. Each HMM in a speech recognizer can be thought of as a generative model of a speech source. A common system for speech recognition is to create an HMM for each word in a finite vocabulary. The problem in designing such a system is the limited speech data for training and testing. If discrete HMMs are used to construct the recognizer, certain parameters of the HMMs will become zero which will cause major errors in recognition. To overcome the problems with the parameters, Zhao and Rowden (1992) proposed using a SOFM to smooth the parameters of the HMMs. Simulations in which the SOFM was used were considered to be using "trained data", whereas simulations in which the SOFM was not utilized were said to be using "untrained data". Without going into details of the HMM model, recognition of trained speech data can be raised from 56.833%, using untrained data, to over 99%. To extend their analysis, Zhao and Rowden used a 3D SOFM and found even better results than with the 2D case. The major advantage with their approach is that

it required less computational time than other HMM parameter smoothing techniques discussed in their report such as Parzen smoothing.

Further ideas and advances in the area of speech recognition should be facilitated with some of the recent work done on the SOFM. For instance, Bauer and Pawelzik (1992) discuss the concept of using the topographic product to measure the degree to which neighborhood relations are preserved from input to output space. They used their approach on a 19-dimensional speech data set and found that a 3-dimensional output space is better suited to the data. This coincides with the improved performance noted by Zhao and Rowden (1992) when they used a 3D SOFM for HMM parameter smoothing. Also, Chappell and Taylor (1993) recently published their work on temporal SOFMs. This novel approach to lend SOFMs the understanding of temporality may improve the ability of the SOFM to handle temporal variation . This is significant, for the inability of neural networks in dealing with temporal variations has limited the application of neural nets in the area of speech recognition.

The final application to be discussed in this report involves semantic recognition. The major difficulty in this research area arises from the fact that the words being analyzed are mere symbolic representations of the contextual interpretation being sought. For instance, how should the metric distances between symbolic items be measured? The key is that during the self-organizing process, both the contextual and symbolic aspects of the items are presented to the network. Letting x_c and x_s represent respectively the contextual and symbolic components of the input vector x, x is defined as:

$$x = \begin{bmatrix} x_s \\ x_c \end{bmatrix} = \begin{bmatrix} x_s \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ x_c \end{bmatrix}.$$

The core idea behind symbol maps is that the two components are weighted such that the norm of the contextual part predominates over that of the symbol part during the self-organizing

process (Kohonen, 1990). This predominance ensures that the symbols are mapped into a spatial order reflecting their semantic similarities. During recognition processes, some or all of the contextual part of the input signal may be missing, but the symbolic signals are active all of the time. Thus, if during recognition of input information, the contextual portions of the signal are weaker or are missing, the same map units are selected solely on the basis of the symbolic part. Ritter and Kohonen (1989) demonstrated this idea using a simple language. The vocabulary consisted of verbs, nouns and adverbs. Each word class is further subdivided into categories such as names of people or animals. In order to study semantics in the purest form, semantic meaning of individual words can only be inferred from the context in which the words occur and not from any pattern used for the encoding of the words (Kohonen, 1990). Therefore, each word

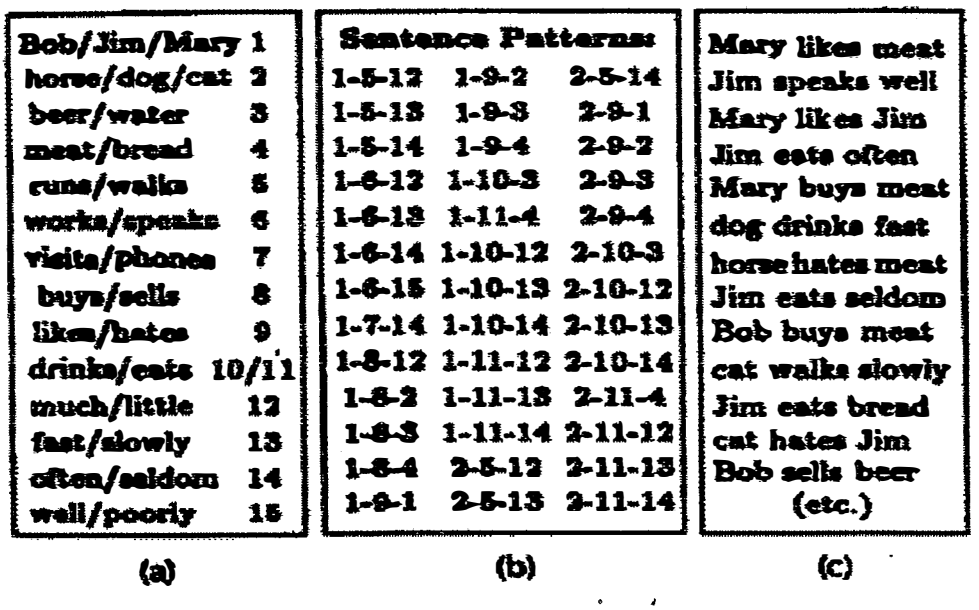

Fig. 7. Outline of vocabulary used for semantic recognition experiment. (a) List of words (nouns, verbs, adverbs), (b) sentence patterns, and (c) some examples of generated three-word sentences. From Kohonen, 1990.

was represented by a random vector of unit length (seven-dimensional). Ritter and Kohonen decided to use randomly generated meaningful three word sentences as input. Figure 8 lists the vocabulary and sentence structure used for the experiment.

For each word, its seven-dimension "symbol field" x_s is simply the code vector of the word itself. The context of a word was defined as the average over 10,000 sentences of all code vectors of predecessor/successor pairs surrounding that word. These 14-dimensional "average word context" vectors, normalized to unit length, represented the "context field" x_c. After 2,000 presentations of corresponding $x = (x_s, x_c)$ combinations were presented to a 10 ×15 output grid, an impressive map resulted (Kohonen, 1990). Nouns, verbs and adverbs were segregated

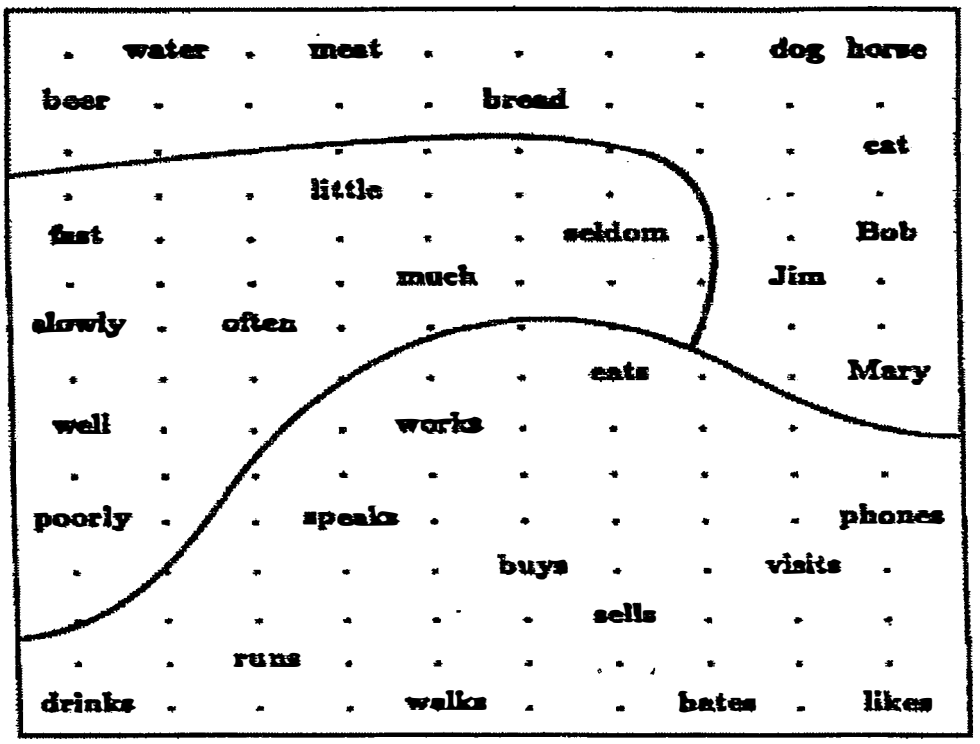

Fig. 8. Semantic map obtained on a network of 10 ×15 cells as described in the text. Nouns, verbs and adverbs are segregated into different domains. Further grouping within each domain is discernible. From Kohonen, 1990.

into different domains. Furthermore, additional grouping with respect to semantic meaning within each domain is easily observable. Figure 8 reflects the results.

One area that would benefit from further work in semantic pattern recognition involves the Query-by-Browsing (QBB) method of querying a database (Beale and Finlay, 1992). Rather than focus on the query itself which traditional query-by-example (QBE) methods do, QBB focuses on the list of required records, that is, the actual query goal. Essentially, the user starts with a listing of the entire database or a subset thereof, marking records of interest while rejecting those of no interest. The system then attempts to guess what the user's criterion is. This "guessing" is further refined by additional communications between the computer and the user. One important aspect of this QBB mechanism is the proper interpretation of the semantics of the user's problem. Thus, advances in semantic recognition may be of importance in developing QBB systems.

Although a few examples of SOFM applications were discussed, many more have been reported. For instance, Ritter et al. (1992) discuss extensively the application of this neural network paradigm to robotics. This wide range of applications of the SOFM demonstrates its versatility. More research is required to uncover further features and properties of the model which should aid in the development of more and different applications.

5) Conclusion

The SOFM paradigm for neural networks has received much attention recently for a number of reasons. First, the self-organizing, unsupervised method of network training is very useful in some applications. Second, the basic model is fairly easy to understand with few

parameters to manipulate. Lastly, the model has found increasing acceptance as an explanation for many biological phenomena such as sensory perception and analysis.

As seen in the preceding discussion, the SOFM is a very efficient and effective model for solving the TSP. As compared to other non-SOFM algorithms, some variants gave good solutions for tour lengths even for problems of considerable size (10,000 cities). These results appear promising but there is a great deal of work yet to accomplish. Of primary importance is further research to discover the mathematical properties of the model. Some work has been done of the mathematical properties of neural computing in general (Amari, 1990; Amari, 1991). Other articles have dealt with the properties of self-organizing maps in particular(Erwin et al., 1992(a); Erwin et al. 1992(b); Lo and Bavarian, 1991; Ritter and Schulten, 1986; Ritter and Schulten, 1988; Yang and Dillon, 1992). Much of the current efforts has been focusing on one-dimensional cases of the SOFM (as used for the TSP). Since it is the higher-dimensional instances of the SOFM which is used in a vast majority of applications, it is necessary that properties of the higher-dimensional cases be investigated.

Many applications of the SOFM model have been proposed. This paradigm lends itself well to applications requiring unsupervised clustering of data vector points. The clustering is usually based on some dominant characteristic of the input set. This characteristic does not have to be known (and often is not) prior to the network training. Since the base model does not actually classify the various resulting clusters, classification (if desired) must be done by either adding a classifying back-end to the trained map (LVQ1) or by direct alterations to the basic model (counterpropagation). For instance, the semantic map on figure 8 reflects a mapping of discernible domains and sub-domains of words. The axis of the map are not important, only the representative input-output signal at each cluster.

Further research into the properties and applications of the SOFM is both necessary and highly justified. Current research has revealed many properties of the model. In addition, a number of useful applications have been designed such as those in the areas of speech recognition and robotics. Also, a couple of recent articles have discussed hardware

implementations of the SOFM model (Macq et al. 1993; He and Cilingirglu, 1993). Thus, the areas for further research are many and varied. Advances in mathematical properties could be pursued. Existing applications could be improved or new applications developed. The development of applications should be particularly important, for this is the real test of the usefulness of the model. For instance, as discussed earlier, advances in semantic recognition could help in the development of relational query by browsing systems. Also, applications in the area of robotics will be useful such as smoothing the movement of robotic arms. Lastly, applications in the areas of medicine or biology could be very beneficial. A possible example is gene sequencing. Whether pursuing theoretical or application development, future research into self-organizing feature maps should be both challenging and rewarding.

6) References

Amari, S. (1990). Mathematical foundations of neurocomputing, *Proceedings of the IEEE*, vol. 78, no. 9, pp 1443-1463.

Amari, S, (1991). Mathematical theory of neural learning, *New Generation Computing*, vol. 8, pp. 281-294.

Aarts, E. and Korst, J. (1989). *Simulated Annealing and Boltzman Machines*, New York: Wiley.

Angeniol, B., de La Croix Vaubois, G., and Le Texier, J-Y. (1988). Self-organizing feature maps and the traveling salesman problem, *Neural Networks*, vol. 1, pp. 289-293.

Banzhaf, W. (1990). The "molecular" traveling salesman, *Biological Cybernetics*, vol. 64, pp. 7-14.

Bauer, H-U. and Pawelzik, K.R. (1992). Quantifying the neighborhood preservation of self-organizing feature maps, *IEEE Transactions on Neural Networks*, vol. 3, pp. 570-578.

Beale, R. and Finlay, J. (1992). Human issues in the use of pattern recognition techniques in *Neural Networks and Pattern Recognition in Human-Computer Interaction*, pp. 429-449, New York: Ellis Horwood.

Bonomi, E., and Lutton, J-L. (1984). Th N-city traveling salesman problem: statistical mechanics and the metropolis algorithm, *SIAM Review*, vol. 26, pp. 551-568.

Brill, F.Z., Brown, D.E., and Martin, W.N. (1992). Fast genetic selection of features for neural network classifiers. *IEEE Transactions on Neural Networks*, vol. 3, pp. 324-328.

Burke, L.I. and Damany, P. (1992). The guilty net for the traveling salesman problem, *Computers and Operations Research*, vol. 19, pp. 255-265.

Burrascano, P. (1991). Learning vector quantization for the probabilistic neural network, *IEEE Transactions on Neural Networks*, vol. 2, pp. 458-461.

Caudill, M. (1993). A little knowledge is a dangerous thing, *AI Expert*, vol. 8, no. 6, pp.16-22.

Chappell, G.J. and Taylor, J.G. (1993). The temporal Kohonen map, *Neural Networks*, vol. 6, pp. 441-445.

Coleman, K.G. and Watenpool, S. (1992). Neural networks in knowledge acquisition, *AI Expert*, vol. 7, no. 1, pp. 36-39.

DeSieno, D. (1988). Adding a conscience to competitive learning, *Proc. Int. Conf. Neural Networks*, vol. 1, pp. 117-124, New York: IEEE Press.

Durbin, R. and Willshaw, D. (1987). An analogue approach to the traveling salesman problem using an elastic net method, *Nature*, vol. 326, pp. 689-691.

Erwin, E., Obermayer, K. and Schulten, K. (1992 (a)). Self-organizing maps: stationary states, metastability and convergence rate, *Biological Cybernetics*, vol. 67, pp. 35-42.

Erwin, E., Obermayer, K. and Schulten, K. (1992 (b)). Self-organizing maps: convergence properties and energy functions, *Biological Cybernetics*, vol. 67, pp. 47-55.

Favata, F. and Walker, R. (1991). A study of the application of Kohonen-type neural networks to the traveling salesman problem. *Biological Cybernetics*, vol. 64, pp. 463-468.

Fogel, D.B. (1988). An evolutionary approach to the traveling salesman problem, *Biological Cybernetics*, vol. 60, pp.139-144.

Fogel, D.B., Fogel, L.J., and Porto, V.W. (1990). Evolving neural networks, *Biological Cybernetics*, vol. 63, pp. 487-493.

Gersho, A. and Shoham, Y. (1984). Hierarchical vector quantization of speech with dynamic codebook allocation, *Proceedings of the IEEE ICASSP 1984 (10.9/1-4)*, New York: IEEE Press.

Grossberg, S. (1982). *Studies of Mind and Brainl*, Boston:Reidel.

He, Y. and Cilingiroglu, U. (1993). A charged-based on-chip adaptation Kohonen neiral network, *IEEE Transactions on Neural Networks*, vol. 4, no. 3, pp. 462-469.

Hecht-Nielsen, R. (1988). Applications of counterpropogation networks, *Neural Networks*, vol. 1, pp. 131-139.

Hoittis, A. (1993). Inside a self-organizing map, *AI Expert*, vol. 8, no. 4, pp. 38-43.

Hopfield, J.J. and Tank, D.W. (1985). "Neural" computation of decisions in optimization problems. *Biological Cybernetics*, vol. 52, 141-152.

Kangas, J.A., Kohonen, T.K., and Laaksonen, J.T. (1990). Variants of self-organizing maps, *IEEE Transactions on Neural Networks*, vol. 1, pp. 93-99.

Kirkpatrick, S., Gelatt, Jr., C.D., and Vecchi, M.P. (1983). Optimization by simulated annealing, *Science*, vol. 220, pp. 671-680.

Kohonen, T. (1981). Automatic formation of topological maps in self-organizing systems, In E. Oja & O.Simula (Eds.), *Proceedings of the 2nd Scandinavian Conference on Image Analysis,* pp. 214-220, Espoo: Suomen Hahmontunnistustutkimuksen Seuro.

Kohonen, T. (1989). *Self-Organization and Associative Memory*, 3rd Ed., Germany: Springer-Verlag.

Kohonen, T. (1990). The self-organizing map, *Proceedings of the IEEE*, vol. 78, pp. 1464-1480.

Kovacic, M. (1991). Markovian neural networks. *Biological Cybernetics*, vol. 64, pp. 337-342.

Lin, S. and Kernighan, B.W. (1973). An effective heuristic algorithm for the traveling salesman problem, *Operations Research*, vol. 21, pp. 498-516.

Lippmann, R.P. (1987). An introduction to computing neural nets, *IEEE ASSP Magazine*, vol.4, pp. 4-22.

Lo, Z. and Bavarian, B. (1991). On the rate of convergence in topology preserving neural networks, *Biological Cybernetics*, vol. 65, pp. 55-63.

Looi, C. (1992). Neural network methods in combinatorial optimization, *Computers and Operations Research*, vol. 19, pp. 191-208.

Macq, D., Verleysen, M., Jespers, P., and Legat, J. (1993). Analog implementation of a Kohonen map with on-chip learning, *IEEE Transaction on Neural Networks*, vol. 4, no. 3, pp. 456-461.

Ritter, H. and Kohonen, T. (1989). Self-organizing semantic maps, *Biological Cybernetics*, vol. 61, pp. 241-254.

Ritter, H., Martinetz, T., and Schulten, K. (1992). *Neural Computation and Self-Organizing Maps, An Introduction*, Reading, Massachusetts: Addison-Wesley.

Ritter, H. and Schulten, K. (1986). On the stationary state of Kohonen's self-organizing sensory mapping. *Biological Cybernetics*, vol. 54, pp. 99-106.

Ritter, H. and Schulten, K. (1988). Convergent properties of Kohonen's topology conserving maps: fluctuations, stability, and dimension selection, *Biological Cybernetics*, vol 60, pp. 59-71.

Simpson, P.K. (1990). *Artificial Neural Sytems, Foundations, Paradigms, Applications, and Implementations,* New York: Pergamon Press.

Tirri, H. (1991). Implementing expert system rule conditions by neural networks, *New Generation Computing*, vol. 10, pp. 55-71.

Wilson, G.V. and Pawley, G.S. (1988). On the stability of the traveling salesman problem of Hopfield and Tank, *Biological Cybernetics*, vol. 58, pp. 63-70.

Xu, X. and Tsai, W.T. (1991). Effective neural algorithms for the traveling salesman problem, *Neural Networks*, vol. 4, pp. 193-205.

Yang, H. and Dillon, T.S. (1992). Convergence os self-organizing neural algorithms, *Neural Networks*, vol. 5, pp. 485-493.

Zhao, Z. and Rowden, C.G. (1992). Use of Kohonen self-organizing feature maps for HMM parameter smoothing in speech recognition, *IEEE Proceedings-F*, vol. 139, pp. 385-390.

School of Business
McMaster University

WORKING PAPERS - RECENT RELEASES

361. Roy J. Adams, "Employment Relations in an Era of Lean Production", April, 1991.

362. John W. Medcof, John Bachynski, "Measuring Opportunities to Satisfy the Needs for Achievement, Power and Affiliation", April, 1991.

363. Robert F. Love, Paul D. Dowling, "A Nested Hull Heuristic for Symmetric Traveling Salesman Problems", June, 1991.

364. Robert F. Love, John H. Walker, and Moti L. Tiku, "Confidence Levels for $\ell_{k,p,\theta}$ Distances", June, 1991.

365. Archer, N. P., and G. O. Wesolowsky, "A Dynamic Service Quality Cost Model with Word-of-Mouth Advertising", July, 1991.

366. John W. Medcof, "Weiner, Kelley, Jones, and Davis, Peat: Integrated", August, 1991.

367. Harish C. Jain and Rick D. Hackett, "A Comparison of Employment Equity and Non-Employment Equity Organizations On Designated Group Representation And Views Towards Staffing", August, 1991.

368. J. Brimberg and R. F. Love, "Local Convergence in a Generalized Fermat-Weber Problem", August, 1991.

369. Rick D. Hackett and Peter Bycio, "The Temporal Dynamics Associated with Absenteeism as a Coping Mechanism", September, 1991.

370. Robert F. Love and John H. Walker, "A New Criterion For Evaluating the Accuracy of Distance Predicting Functions", October, 1991.

371. Joseph B. Rose, "Interest and Rights Disputes Resolution in Canada, New Zealand and Australia", October, 1991.

372. Paul A. Dion and Peter M. Banting, "Buyer Reactions to Product Stockouts in Business to Business Markets", October, 1991.

373. Isik U. Zeytinoglu, "Part-Time and Other Non-Standard Forms of Employment: Why are They Considered Appropriate for Women?", November, 1991.

374. Rick D. Hackett, Peter Bycio and Peter Hausdorf, "Further Assessments of a Three-Component Model of Organizational Commitment", January, 1992.

375. Y. Lilian Chan, Bernadette E. Lynn, "Evaluating Tangibles and Intangibles of Capital Investments", February, 1992.

376. Christopher K. Bart, "Strategic Lessons for Today's Airline CEO's", February, 1992.

377. Yufei Yuan, "A Stable Residence Exchange Problem", March, 1992.

378. Jim Gaa, "The Auditor's Role: The Philosophy and Psychology of Independence and Objectivity". April, 1992.

379. David J. Buchanan and George O. Wesolowsky, "Algorithm for Rectilinear Distance Minimax Single Facility Location in the Presence of Barriers to Travel", April, 1992.

380. Thomas E. Muller and D. Wayne Taylor, "Eco-Literacy Among Consumers: How Much Do They Know About Saving Their Planet?", April, 1992.

381. Willi H. Wiesner, "The Contributions of Job Relevance, Timing, and Rating Scale to the Validity of the Structured Employment Interview", September, 1992.

382. Robert G. Cooper, "Third Generation New Product Processes", November, 1992.

383. Robert F. Love and John H. Walker, "Distance Data for Eighteen Geographic Regions", August, 1993.

384. Jack Brimberg and Robert F. Love, "New Hull Properties for Location Problems", August, 1993

385. Harish C. Jain and S. Muthuchidambaram, "Bill 40 Amendments to Ontario Labour Relations Act: An Overview and Evlauation", September, 1993.