Combinatorial Optimization Using a Continuous State Boltzmann Machine

Kurt M. Gutzmann
Planning Research Corporation
1500 Planning Research Drive
McLean, VA 22102

ABSTRACT

The non-Euclidean NP-complete traveling salesman problem (TSP) is solved heuristically using a new continuous state Boltzmann machine and the usual binary state Boltzmann machine. The TSP is formulated with a nonlinear objective function and linear constraints. The LaGrangian function for the formulation is shown to be equivalent to the energy function of a Boltzmann machine. The formulation of the interconnection weight matrix for the Boltzmann machine follows directly from the LaGrangian. The continuous state Boltzmann machine locates tours of comparable shortness but with much greater searching efficiency than the binary state Boltzmann machine. The effects of two kinds of annealing schedules, linear and exponential decay, and the number of update cycles are also examined. The results indicate that the discriminatory power of Boltzmann machines is significant. These highly fault-tolerant neural networks can solve NP-complete problems with an interconnection network density that scales as the reciprocal of the size of the problem.

1. Introduction

Networks of simple computational units operating in parallel can collectively provide powerful computing capabilities. Interest in the organization of computational networks and their properties has increased as the limitations of serial computers become more evident.

The traveling salesman problem (TSP) is of interest because it is NP-complete (Papadimitriou, 1977), so that heuristics suitable for neural computation of high quality approximate solutions are of value. Other problems in the NP-complete category are the Boolean satisfiability problem wherein the values of Boolean variables that cause an expression to be true are sought, and the bin-packing problem. These problems are related to each other by polynomial-time transformations, so that a neural heuristic that works for one problem might be applied to other problems in the set NP.

In this paper a continuous state Boltzmann machine is used to locate optimal and near-optimal solutions to the (TSP). The continuous state machine is compared through simulations with the usual discrete state Boltzmann machine. The process used by the Boltzmann machine to locate energy minima is known as simulated annealing (Metropolis et al., 1953; Kirkpatrick et al., 1983). The TSP is formulated as a nonlinear zero-one program with equality constraints. The LaGrangian is then mapped directly to the Boltzmann machine interconnection weight matrix. The effects of annealing schedule and number of update cycles on solution quality and effort are examined.

An interesting work on neural computation of solutions for the planar TSP is by Hopfield et al. (1985). Hopfield and Tank devised a deterministic analog circuit neural network for solution of optimization problems, and successfully demonstrated its application to the TSP. Hopfield's neural network obtained solutions by relaxing to energy minima in a deterministic manner. Good solutions were obtained for problems of 10 cities and in one instance for a problem for 30 cities. The network was "programmed" by judicious selection of the circuit parameters and topology. The energy function used was developed from *ad hoc* principles based on the investigator's desired performance of the network. The energy function was of the form E = A(row inhibitions) + B(column inhibitions) + C(exactly n units on) + D(tour length). Searching for the parameters A,B,C, and D was performed in order to get the network to operate correctly. In the 10 city problems the network located solutions approximately 75% of the time. The other 25% of the time the solutions did not correspond to permutation matrices, and this same behavior will be observed in the Boltzmann machines presented in this paper. Currently AT&T Bell Laboratories is producing experimental VLSI circuits which implement this kind of neural network. Similar stochastic networks can serve as a content addressable memory (CAM) (Hopfield 1982), where retrieval is accomplished by setting bits to the recall key and allowing the network to relax to a stable state. The recall process is essentially that of determining which of the stored words is closest (in Hamming distance) to the key word.

Selman (1985) used a Boltzmann machine for parsing language described by a context-free grammar. The low energy states of the machine corresponded to the correct parse of input applied to the 'input' units of the machine. Selman's parsing scheme was based on connectionist primitives representing grammar productions. The primitives were linked so that individual rules could be selected from a collection of rules. Selman used a machine with states of plus and minus one (as opposed to one and zero) to obtain symmetrical energy relationships which aided in the selection of thresholds and weights to describe a grammar.

Simulated annealing for combinatorial optimization was applied successfully on traditional serial computers by Kirkpatrick et al. (1983). Press et al. (1986) offered a simulated annealing procedure for obtaining excellent, and occasionally optimal, solutions for the TSP on serial computers. Solutions obtained to a few test problems using the procedure of Press were compared by the author with solutions generated by three other heuristic methods: the space filling curve of Bartholdi et al. (1983 and 1982) the cell order method of Iri et al. (1982 and 1983) and the strip method of Supowit et al. (1983). One drawback of these three heuristics is that they are applicable only to the planar Euclidean TSP. The simulated annealing process was consistently able to locate better solutions from a random starting solution as well as improve suboptimal solutions generated by the other three heuristics, but a more detailed study of the properties of these heuristics is required to assess accurately the relative merits of each.

Szu (1986a,1986b) used a Cauchy probability distribution in order to generate occasional long moves in the search space for combinatorial optimization problems. Szu applied the simulated annealing with Cauchy probability distribution, termed fast simulated annealing, to a tracking and correlation problem of obtaining accurate estimates of the ground tracks of moving emitters based on bearing-only data from several sensor sites.

Simulated annealing offers a powerful search heuristic for obtaining excellent solutions for problems in the *NP-complete* category like the TSP (Papadimitriou 1977) Application of simulated annealing in a neural computation provides even faster solution of complex problems like the TSP, as will be shown in this paper.

2. The Boltzmann Machine

The Boltzmann machine has been described as a parallel constraint satisfaction network that learns (Ackley et al., 1985; Hinton et al., 1984), but more fundamentally it is an energy minimization machine. While the domain-independent learning capabilities of the Boltzmann machine provide a powerful tool for use in artificial intelligence and machine learning problems (Gutzmann, 1986), we are interested here in constructing machines that can solve difficult optimization problems in parallel, without the requirement of the lengthy learning process.

A Boltzmann machine is composed of a fully connected network of simple units which appear probabilistically in one of two states, *on* or *off*. The links or connections among the units are symmetric and have associated weights, w_{ij}. The weights may have real values of either sign. The individual units of the Boltzmann machine are likened to elemental hypotheses about the machine's domain. A unit in the on state indicates that an elemental hypothesis is considered true, and the off state indicates that an elemental hypothesis is considered false. The link weight corresponds to a weak constraint among a pair of elemental hypotheses, where positive weight indicates mutually supporting hypotheses, and negative weight indicates mutually contradicting hypotheses.

The global state of this system can be assigned a single number, referred to as the energy of that state. The individual units can be made to minimize the energy of the system. The energy, *E*, for a Boltzmann machine composed of *m* units is defined as

$$E = -\sum_{i=1}^{m}\sum_{j=i+1}^{m-1} w_{ij}s_i s_j \qquad + \sum_{i+1}^{m} \theta_i s_i \tag{1}$$

with w_{ij} the connection weight between units i and j, s_i is 1 if unit i is on and 0 otherwise, and θ_i is a threshold.

The energy function clearly counts the contributions from units that are *on* (the threshold terms), but the role of the weight terms is less obvious. Since each unit corresponds to some elemental hypothesis, a pair of hypotheses that are mutually contradictory (i.e., their link weight is negative) should reflect a higher energy state when both units are *on*. Conversely, mutually reinforcing hypotheses (positive link weight) should reflect a lower energy state. Units that are not interacting directly (not both units *on*) have no influence on the energy. This is the effect of the first term in (1).

The Boltzmann machine is designed to locate energy minima which are then interpreted as solutions to a problem. Each unit can locally determine the change in the global energy of the system due to a change in the local unit's state. This energy difference for the k^{th} unit is obtained in the following manner. Let E_k be the energy contribution of unit k and evaluate E_k with unit k off and on, so that the energy difference is

$$\Delta E_k = \sum_{j=1}^{m} w_{kj}s_j - \theta_k \tag{2}$$

In a deterministic machine this permits a simple rule to govern the state of a unit in order to minimize the global energy. If the energy of its neighbors (defined as the sum of the weights of the 'on' units) exceeds a threshold, then its state is on; otherwise, its state is off. Hopfield (1982) has shown that such a system always relaxes into a local minima.

The terms involving θ in (1) and (2) can be eliminated by assuming the presence of a permanently *true* unit in the system, say unit m is permanently *on*. The expression for E is then

$$E = -\sum_{i=1}^{m-1} \sum_{j=i+1}^{m} w_{ij}s_i s_j \tag{3}$$

The energy difference for the k^{th} unit is developed as before

$$\Delta E_k = \sum w_{kj}s_j \tag{4}$$

Local energy minima by introducting a random component. The unit's state is on with a probability defined by the energy difference of its on and off states and a parameter, T, which is likened to temperature. The state of unit k is set to *on* with probability

$$p_k = 1 / (1 + e^{-\Delta E_k / T}) \tag{5}$$

The probability in (5) varies from 0 to 0.5 for $\Delta E > 0$ and from $.5$ to 1.0 for $\Delta E \leq 0$. The relative probability of two global states α and β has a Boltzmann distribution as given in (6) below, and hence the name for this machine. This indicates that the equilibrium distribution of states in the machine is independent of the path to that state.

$$P_\alpha / P_\beta = e^{-(E_\alpha - E_\beta)/T} \tag{6}$$

At low temperatures the machine is biased strongly to states of low energy, whereas at high temperatures there is not such a strong bias. An interesting aspect of the Boltzmann machine is that a domain independent learning algorithm has been formulated for it by Ackley et al. (1985). The learning algorithm modifies the connection strengths of the machine so that the entire network develops an internal model of the structure of its environment. The learning algorithm manipulates the probabilities of various states by modifying the weights.

The application of Boltzmann machines presented here does not involve the learning algorithm, but instead seeks to find a direct relationship between the objective function and constraints of an optimization problem and the weight set of a Boltzmann machine.

3. The Continuous State Boltzmann Machine

The continuous state Boltzmann machine is introduced here as a variation on the discrete Boltzmann machine. In the continuous machine the states of units are in the range [0,1], instead of being simple binary units. If the binary machine is viewed as using Boolean logic among its hypotheses, the continuous machine might be viewed as using fuzzy or real-valued logic among its hypotheses. This allows the continuous machine to simultaneously evaluate several alternative states.

The selection of a new state value in the updating algorithm is based on the truncated exponential distribution. The truncated exponential distribution is parameterized by temperature and the energy difference (2,4). The truncation points of the distribution are 0 and 1. The density function for an exponential distribution truncated at a and b $(a<b)$ is $f(x)=\lambda e^{-\lambda x}/(e^{-\lambda a}-e^{-\lambda b})$. Since the truncation points for the continuous machine are 0 and 1,

$$f(x)=\lambda e^{-\lambda x}/(1-e^{-\lambda}) \tag{7}$$

and the distribution function is

$$F(x)= (1-e^{-\lambda x})/ (1-e^{-\lambda}) \tag{8}$$

where $\lambda = |\Delta E|/T$. The update algorithm for a unit in the continuous Boltzmann machine is to set the state of the unit to a random deviate from a truncated exponential distribution with parameter λ defined above if $\Delta E \leq 0$ and to set the state of the unit to the complement of a deviate from a truncated exponential distribution if $\Delta E > 0$. As T approaches zero the truncated exponential distribution forms a Dirac delta function with all of its probability concentrated at zero. Using the updating algorithm above, the limiting form of the continuous state Boltzmann machine is a binary state machine as the temperature approaches zero.

4. Formulation of the TSP for Boltzmann Machines

The TSP is one of making a minimal cost tour among n cities (numbered 1 to n). The tour must visit each city only once and return to its starting point. The cost of traveling from city k to city l is D_{kl}, and the cost matrix D may be nonsymmetric; i.e., D_{kl} is not necessarily equal to D_{lk}. The minimum-weight matching problem (Bartholdi et al., 1983; Iri et al., 1983; Supowit et al., 1983) is closely related to the TSP.

For convenience in mapping the problem to a Boltzmann machine composed of n^2 units as shown in Figure 1, the variable S_i is introduced. S_i is a *zero-one* variable and is related to a city and its position in the tour by the following functions:

$$r(i) = 1 + \lfloor (i-1)/n \rfloor \tag{9}$$

$$c(i) = 1 + (i-1) \bmod n \tag{10}$$

where $r(i)$ gives the index of the city (or row in the Boltzmann machine) and $c(i)$ gives the position of the city in the tour (or column in the Boltzmann machine). $S_i = 1$ if city $r(i)$ is in the $c(i)^{th}$ position in the tour (i.e. it is the $c(i)^{th}$ city visited) and is 0 otherwise. When S is arranged in a matrix as shown in Figure 1, a valid tour will be a permutation matrix, having exactly one 1 in each row and column.

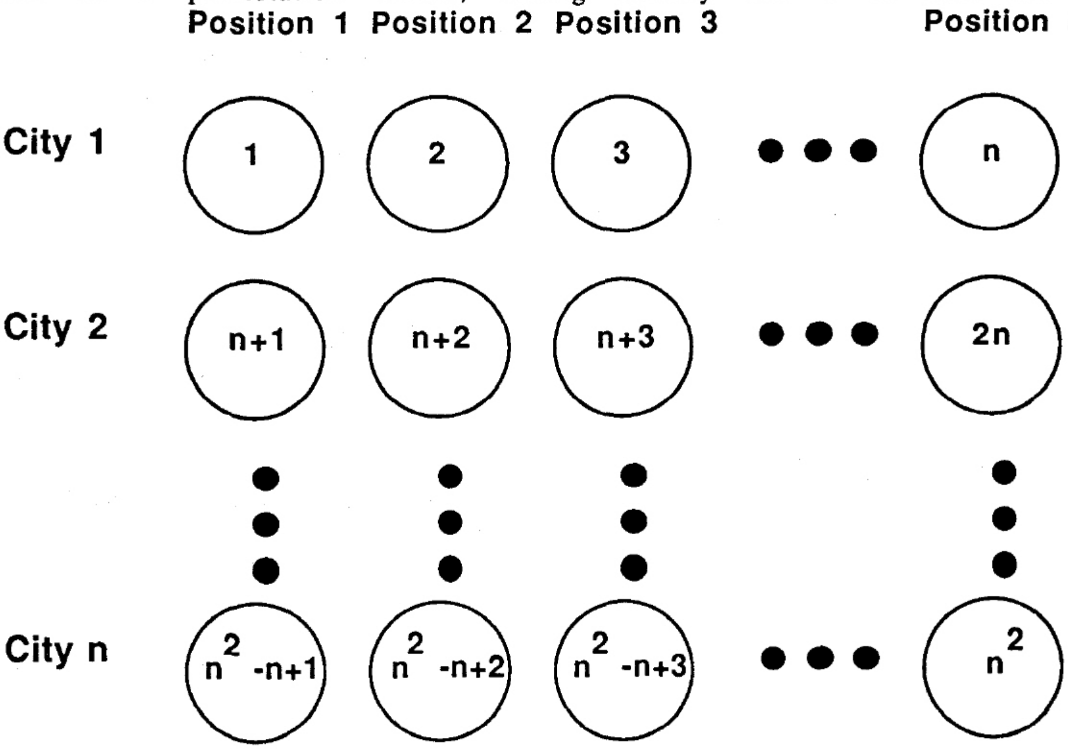

Figure 1 : Boltzmann Machine Layout for TSP

The objective function is based on the pairs of $S_i S_j$ equal to one. The pairs contribute to the objective function only if they are adjacent on the tour; otherwise, the contribution is zero. This might lead us to minimize an objective function with $S_i S_j D_{r(i),r(j)}$ terms in it, but this poses a special problem for a Boltzmann machine. The weight terms between the units would be negative to inhibit the adjacent placement of cities in the route with large D and to encourage the adjacent placement of cities with small D. Similarly, negative weights are used to enforce the constraints on producing permutation matrices as solutions. The problem is that this would permit the minimum energy state to be zero with all units off (or no interacting pairs of units on). However, by making the sign of the weights modeling the intercity travel costs positive, the machine can obtain energies less than zero by turning on a pair of units for adjacent cities. The constraints of the problem are still modeled with negative (inhibitory) weights, so a positive term in the energy function (which is preceded by a minus sign) to model the intercity travel costs will provide the desired behavior. If D_{max} is the largest value in D, an objective function which when minimized (subject to constraints) provides an optimal length tour is

$$min\ f(S) = -\Sigma S_i S_j (D_{max} - D_{r(i),r(j)})$$ for adjacent pairs of cities in the tour.

This objective function can be described with the use of a function $G(i,j)$ which accounts for the order of travel between the cities and their adjacency in the tour. $G(i,j)$ is zero when the cities indexed by i and j (through the function r, (9)) are not adjacent in the tour, but returns the appropriate value from the distance matrix to account for their order in the tour when they are adjacent.

$$G(i,j) = \begin{cases} D_{max} - D_{r(i),r(j)} & c(i)c(j)=1 \text{ or } c(i)=c(j)=n-1 \\ D_{max} - D_{r(j),r(i)} & c(i)\text{-}c(j)=1 \text{ or } c(i)=c(j)=1-n \\ 0 & otherwise \end{cases} \qquad (11)$$

This leads to the following formulation for the TSP:

$$min \qquad f(S) = \sum_{i=1}^{n^2-1} \sum_{j=i+1}^{n^2} S_i S_j G(i,j) \qquad (12)$$

subject to

$$\sum_{i=1}^{n} S_{(ni-j+1)} = 1 \qquad j=1,2,...,n\ (columns) \qquad (13)$$

$$\sum_{j=1}^{n} S_{(ni-j+1)} = 1 \qquad i=1,2,...,n\ (rows) \qquad (14)$$

$$S_i = 0 \text{ or } 1 \qquad i=1,2,...,n^2$$

The energy function of a Boltzmann machine (1,3) is developed from the interunit connection weights w_{ij}. These interunit weights can be directly related to the TSP formulation above by forming the LaGrangian function $L(S,\lambda)$ which brings the constraints into the objective function with LaGrange multipliers λ_k, $k=1,2,..,2n$. After some algebraic manipulations we have

$$\min \qquad L(S,\lambda) = \sum_{i=1}^{n^2-1} \sum_{j=i+1}^{n^2} S_i S_j G(i,j) \qquad + \sum_{i=1}^{n^2} S_i(\lambda_{r(i)}+\lambda_{c(i)+n})$$

$$+ \sum_{i=1}^{2n} \lambda_i \tag{15}$$

This has the form of the energy function of a Boltzmann machine where $G(i,j)$ provides the interunit weights and the LaGrange multipliers operate as threshold terms, except for the final term involving the sum of the LaGrange multipliers. In minimizing $L(S,\lambda)$ we would ordinarily seek values for S and λ by solving the partial derivatives of $L(S,\lambda)$ for zeros and checking the sufficiency conditions for a minimum (using the bordered Hessian, for example). This approach is not feasible with this system since we have a zero-one variable problem and non convex objective function.

In solving the TSP with a Boltzmann machine, we are performing a parallel, simultaneous search for energy minima which correspond to solutions of the TSP. It is possible to formulate the interconnection weight matrix of the machine so that the thresholds corresponding to the sums of pairs of LaGrange multipliers (and the multipliers themselves) can be made zero and the energy minima of the machine correspond to optimal tours.

Suppose that we have a Boltzmann machine in one of its global energy minima states with energy E^* as shown in Figure 2, where the *on* units are shaded black. Only active arcs between units are shown in the figure. Let us assume that the thresholds of the Boltzmann machine are zero (i.e., all the λ are zero), so that if unit l (shaded gray) changes its state to *on* the change in the energy will be $\Delta E = +2q - (D_{max} - D_{r(i),r(j)}) - (D_{max} - D_{r(j),r(k)})$, and will be at most $2q-2D_{max}$ when $D_{r(i),r(j)}=D_{r(j),r(k)}=0$. This same analysis holds if unit l is turned *on* and some other unit becomes a candidate to turn *on* as before. The inhibitory weight q must be strictly less than $-D_{max}$ to ensure that the energy minima of the machine correspond to valid tours, when we use a single value for the inhibitory weights. Pairs of LaGrange multipliers λ determine the thresholds in the energy function, so that when they are set at zero the inhibitory weight q must be less than $-D_{max}$. The energy function of (3) is now of the same form as the LaGrangian function of (15). The function G (11) is modified slightly to accommodate the inhibitory weight of $-D_{max}$, and will be denoted G' as follows:

$$G'(i,j) = \begin{cases} D_{max}-D_{r(i),r(j)} & c(i)-c(j)=-1 \text{ or } r(i)=c(j)=n-1 \\ D_{max}-D_{r(j),r(i)} & c(i)-c(j)=1 \text{ or } c(i)=c(j)=1-n \\ -D_{max} & c(i)=c(j) \text{ or } r(i)=r(j) \\ 0 & \text{otherwise} \end{cases} \tag{16}$$

The function G' provides the definition of the interconnection weight matrix for the Boltzmann machine. The global minima of the energy function of a Boltzmann machine defined as

$$E = -\sum_{i=1}^{n^2-1} \sum_{j=i+1}^{n^2} S_i S_j G'(i,j) \qquad (17)$$

correspond to optimal tours for the TSP.

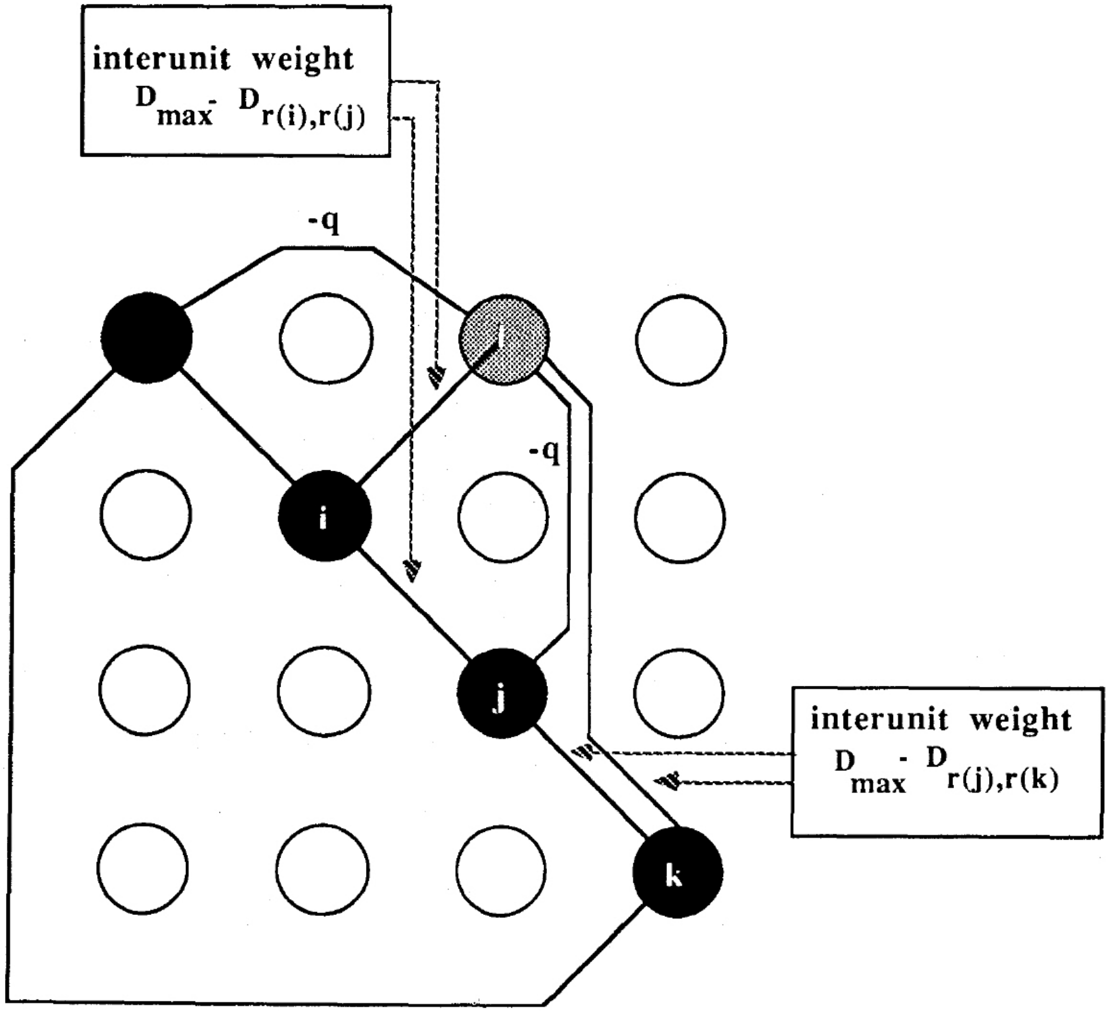

Figure 2 : Effect of Inhibitory Weights on Energy

5. Simulation of Boltzmann Machines Solving the TSP

A simulation experiment was carried out to test the abilities of the continuous and discrete Boltzmann machines to locate good solutions to the TSP. The number of updates each unit received during an annealing and the annealing schedule were also part of the experimental design. The simulation of a Boltzmann machine is relatively straightforward, according to the update algorithms previously discussed in Sections II and III. Throughout the annealing process, units are updated randomly and asynchronously. The update process can be viewed as occurring with some mean rate, although in a serial computer simulation of the Boltzmann machine no simultaneous events are permitted to occur.

Two annealing schedules were used for the simulations, a linear schedule and a negative exponential schedule. The parameters of the schedules are *u*, the number of updates per unit; T_s, the starting temperature; and T_f, the final temperature. For the linear schedule, the temperature *T* at the k^{th} unit update is $T=TS-k\times(TS-Tf)/(un^2)$, where *n* is the number of cities again. For the exponential decay schedule the temperature *T* at the k^{th} unit update is $T=T_s\times exp(k\times ln(T_f/T_s)/(un^2))$. These schedules reduce the temperature incrementally for each update of the machine in a smooth manner. A variety of other schedules are possible, but these were chosen for their simple characteristics and smooth change.

The average number of updates, *u*, that each unit experiences is a measure of time for the Boltzmann machine. One unit of time for a Boltzmann machine corresponds to the time required for each unit to receive one update (on average). Thus *u* may also be considered the total time the machine is operating.

A test problem for $n=10$ cities was generated with uniformly distributed intercity distances between *1* and *10*. The optimal tour length was *10.00*. Knowledge of the optimal tour is used for comparison purposes. The distance matrix generated was asymmetric, so that direction of travel enters the problem as well as order of visitation. Ten cities is a small enough problem to allow a sufficient sample to be generated via simulation (due to processing time on a serial computer), while still providing *3,628,800 (10!)* possible solutions to the problem. The large number of possible solutions is contrasted with the larger number of possible states, 2^{100}, of the *100* units. This variety provides a good test of the Boltzmann machine's ability to discriminate good solutions from meaningless states and poor solutions.

Figure 3 shows the experimental design. This design permits a three-way analysis of variance. The three treatments are number of updates per unit (five levels, *10, 25, 50, 100, 200*), linear or exponential schedule type, and continuous or discrete machine type.

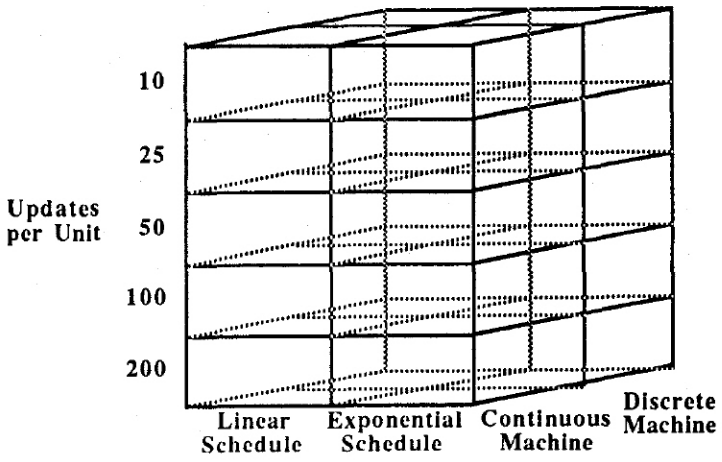

Figure 3 : Three Way Analysis of Variance Experiment Design

For the continuous machine the starting temperature for both schedules was *1.0*, and the ending temperature for both schedules was *0.1*. The continuous machine states were thresholded at *0.5* to determine the final state. The temperatures for the discrete machine schedules were twice those of the continuous machine schedules. This is because, for a given energy difference, the probability of assuming a state interpreted as *on* should be the same *for either machine type. From (5) and (8)* it is clear that the equiprobable temperature for the discrete machine must be twice that of the continuous machine when thresholding for the continuous machine occurs at *0.5*. The row/column inhibitor was set at $-D_{max} \times 1.001$.

Test runs of the simulation indicated that final states corresponding to solutions resulted only part of the time. Therefore, the simulations were run until 50 successes were obtained for each treatment. The equal sample sizes are necessary to avoid nonorthogonality in the sums of squares used in the analysis of variance. The fraction of success and the number of updates per unit for each treatment also provide a measure of the cost of obtaining a solution.

6. Simulation Results

The results of the simulation experiment are shown in Table 1. The table gives the average tour length obtained, the standard deviations of the tour length, the fraction of time a tour was located, and the average computational cost of a solution. Each cell in the table is based on 50 tours located.

Table 1 : Simulation Experiment Results

Updates per Unit Machine/Schedule	10	25	50	100	200
Continuous/Linear					
Average	25.8	23.0	21.5	17.6	16.6
Std.Dev.	6.8	5.5	5.1	5.1	5.4
Pr[success]	0.29	0.20	0.45	0.69	0.69
Cost*	34	125	110	144	288
Discrete/Linear					
Average	30.8	24.9	20.8	17.4	14.9
Std.Dev.	8.5	7.7	6.9	5.2	5.3
Pr[success]	0.15	0.14	0.24	0.29	0.39
Cost*	65	173	208	348	508
Continuous/Expon					
Average	28.8	24.5	22.6	19.8	18.8
Std.Dev.	7.3	6.4	5.5	5.5	4.8
Pr[success]	0.25	0.30	0.31	0.49	0.59
Cost*	40	85	161	204	340
Discrete/Expon.					
Average	32.1	30.4	24.6	21.8	20.3
Std.Dev.	9.0	7.6	7.8	6.5	5.0
Pr[success]	0.15	0.14	0.14	0.15	0.20
Cost*	67	180	347	662	996

** cost is computed as (updates per unit)(number of units)/(fraction of success)*

Sample size in each cell is 50 observations.

The computational cost is developed as (number of units)×(number of updates per unit)/Pr[success]. The expected number of annealings to obtain a valid tour is the reciprocal of the probability of success, since these are Bernoulli trials for this purpose.

As a precursor to the analysis of variance, Cochran's test for the equality of variances was performed on the sample variances. The computed statistic was not significant at the .01 level, so that the assumption of homoscedasticity in the analysis of variance is justified.

Table 2 shows the summary of the three-way analysis of variance. The significance level used is *0.01*. Each of the main effects is significant. The largest mean square is for the updates per unit, indicating that this factor has the most influence on average tour length obtained. Schedule type and machine type are also significant. The updates/machine type interaction and the schedule/machine type interaction are also significant, whereas the updates/schedule interaction is not. The three-way interaction was not significant. Figure 4 shows the average tour length obtained as a function of computational cost for each of the arrangements of machine type and schedule. The ordinate axis has the average tour length and the fraction of tours with less than a certain length. For example, of *10!* tours, *10!*0.00311398* of them have length less than or equal to *30*. These fractions were obtained by enumeration of all tours for the test problem.

Table 2 : Analysis of Variance

Three Way Analysis of Variance Summary

Source of Variation	Sum of Squares	Degrees of Freedom	Mean Square	Computed F	
Main Effects					
Updates per Unit	18195.50	4	4548.62	107.8	*
Schedule Type	2383.10	1	2383.10	56.5	*
Machine Type	924.19	1	924.19	21.9	*
Two Factor Interaction					
Update/Schedule	103.04	4	25.76	.61	
Update/Machine	779.27	4	194.82	4.62	*
Schedule/Machine	294.45	1	294.45	6.98	*
Three Factor Interaction	255.39	4	63.85	1.51	
Error	41354.53	980	42.20		
Total	64288.48	999			

** significant at the 0.01 level*

$F_{0.01}(1,980)=6.63$ $F_{0.01}(4,980)=3.32$

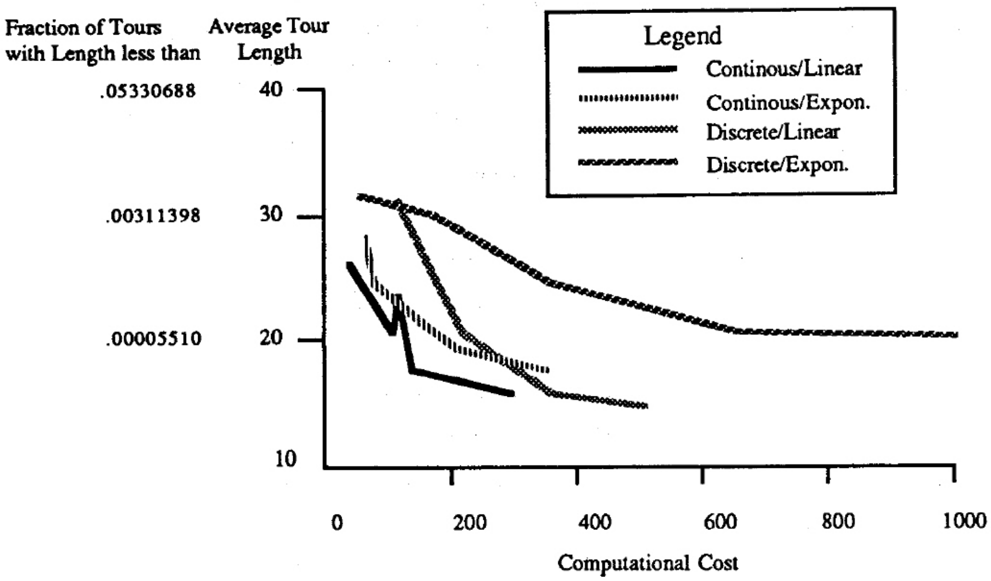

Figure 4 : Solution Quality versus Computational Cost

7. Conclusions

Figure 4 clearly indicates that the searching efficiency of the continuous Boltzmann machine is superior to that of the discrete machine. Although the average tour lengths obtained by either machine are comparable, the much higher probability of success for the continuous state machine gives it the advantage. In either case, the linear schedule performs better in terms of tour length and probability of success.

The number of updates per unit has the largest influence on the length of the tour produced. In the cases with *200* updates per unit, the average tours using the linear schedule are among the 40 best tours out of *3.6* million possibilities. This indicates that the Boltzmann machine is capable of very fine discrimination in highly irregular energy landscapes typical of NP-complete problems.

The continuous machine updating algorithm guides the machine toward energy minima based on the local information available to each unit, but the continuous nature of the units tends to smooth out the energy landscape considerably so that the global energy minima are located with fewer updates of the units. This permits gradual change to a new solution through energy barriers that are not so steep. This continuous space search moves smoothly among states, simultaneously considering several alternative solutions. As the temperature lessens, one alternative crystallizes and the others vanish.

A relatively small number of units in a Boltzmann machine develop considerable computational power. The machine used here also has considerable fault tolerance. There are *n* optimal tours, each with a different starting point but describing the same circuit. The permutation matrices describing these tours are horizontal rotations of each other, so that for a failure mode of 'stuck off' for the units (a typical failure mode for real neurons), an *n*-fold redundancy exists at the unit level.

Ordinarily full interconnection among the units is assumed for Boltzmann machines, requiring a number of connections (or non-zero entries in the weight matrix) equal to the square of the number of units or n^4 connections. However, the weight matrix defined by G' (16) is sparse, containing $(3n\text{-}2)n^2$ non-zero elements out of a total of n^4 elements. The density of the Boltzmann machine interconnection network is the ratio $(3n\text{-}2)/2n^2$, since the links are symmetric and bidirectional. For $n=10$ the connection density is *0.14*, while for $n=1000$ the connection density is only *0.015*.

This indicates that excellent solutions to NP-complete problems can be provided at low computational cost by continuous state Boltzmann machines, with a number of units proportional to the square of the problem size and an interconnection network density that scales as the reciprocal of the size of the problem. However, this solution technique is not expected to scale up very well to larger problems in the case of the TSP. The reason for this is that there are *n!* states corresponding to solutions out of a possible 2^n states. The probability that any randomly generated state is a solution to the TSP is P(solution) = $n! \ / \ 2^{n \times n}$ whose limit is zero as *n* grows without bound. For the case of $n=10$, P(solution) = 2.8×10^{-24},while for $n=30$, P(solution) = exp(-549.17), a probability that for practical purposes is zero. Simulations of the continuous state Boltzmann machine obtained a solution for a problem of *30* cities, but the number of replications before obtaining a solution was large. While the ability of the Boltzmann machine to locate solutions is very good for *n* up to about *16* or *20*, larger problems incur an excessive computational cost.

References

Ackley, D., Hinton, G., Sejnowski, T.: A learning algorithm for Boltzmann machines. *Cognitive Science*, **9**, No. 1147-169 (1985)

Bartholdi, J.J., Platzmann, L.K.: A fast heuristic based on space filling curves or minimum weight matching in the plane. *Info. Proc. Letters*, **17**, 177-180 (1983)

Bartholdi, J.J., Platzmann, L.K.: An O(N log N) lanar travelling salesman heuristic based on space filling curves. *Oper. Res. Letters*, **1**, No. 4, 121-125. (1982)

Gutzmann, K.M.: Computational applications of Boltzmann machines and energy minimization processes. In *Proceedings of the 54th Military Operations Research Society(MORS) Symposium*, Ft. McNair, Wash. D.C., June 24-26, 1986

Hinton, G., Sejnowski, T., Ackley, D: Boltzmann machines: constraint satisfaction networks that learn. Technical Report CMU-CS-84-119, Carnegie-Mellon University, Pittsburgh, PA, 1984

Hopfield, J.J.: Neural networks and physical systems with emergent collective computational abilities. *Proc. Natl. Acad. Sci. USA*, **79**, 2554-2558 (1982)

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

Iri, M., Murota, K., Matsui, S.: An approximate solution for optimizing the plotter pen movement. In: R.F. Drenick and F. Kozi (eds), *Lecture Notes in Control and Information Sciences*, **38**, 572-580 New York: Springer 1982

Iri, M., Murota, K., Matsui, S.: Heuristic for minimum weight perfect matchings. *Networks*, **3**, 67-92 (1983)

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

Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H., Teller, E.: Equation of state calculation by fast computing machines. *Journal of Chemical Physics*, **21**, 1087 (1953)

Papadimitriou, C.H.: The Euclidean travelling salesman problem is NP-complete. *Theoret. Comp. Sci.*, **4**, 237-244 (1977)

Press, W.H., Flannery, B.P., Teukolsky, S.A., Vetterling, W.T.: *Numerical Recipes*. New York: Cambridge University Press, 1986

Selman, B.: Rule-based processing in a connectionist system for natural language understanding. Technical Report CSRI-168, Computer Systems Research Institute, University of Toronto, Canada, 1985

Supowit, K., Reingold, E., Plaisted, D.: The Traveling salesman problem and minimum matching in the unit square. *SIAM J. Comput.* , **12**, 144-156 (1983)

Szu, H., "Non-Convex Optimization," SPIE Vol. 698 Real Time Signal Processing IX (1986a), pp. 59-65.

Szu, H., "Fast Simulated Annealing," American Inst. of Physics Conference Proceedings 151 (1986b) Neural Networks for Computing, John S. Denker:editor, pp. 420-425.