October 1989
LU TP 89-19

"Teachers and Classes" with Neural Networks

Lars Gislén, Carsten Peterson and Bo Söderberg

Department of Theoretical Physics, University of Lund
Sölvegatan 14A, S-22362 Lund, Sweden

International Journal of Neural Systems **1**, 167 (1989)

Abstract:

A convenient mapping and an efficient algorithm for solving scheduling problems within the neural network paradigm is presented. It is based on a reduced encoding scheme and a mean field annealing prescription, which was recently successfully applied to TSP.

Most scheduling problems are characterized by a set of hard and soft constraints. The prime target of this work is the hard constraints. In this domain the algorithm persistently finds legal solutions for quite difficult problems. We also make some exploratory investigations by adding soft constraints with very encouraging results. Our numerical studies cover problem sizes up to $O(10^5)$ degrees of freedom with no parameter tuning.

We stress the importance of adding self-coupling terms to the energy functions which are redundant from the encoding point of view but beneficial when it comes to ignoring local minima and to stabilizing the good solutions in the annealing process.

1 Introduction

Having computers that are able to make logical inferences without exploring a combinatorial space is an attractive and challenging goal. This is a field where humans presently outperform computers despite the relative slowness of the human elementary processor ($O(100ms)$). The reason for the human success in this field is very likely due to a massively parallel processing of a compact data structure. Such a processing paradigm is aimed at by a neural network formulation. This approach has shown great promise in the areas of feature recognition, content-addressable memories and for solving optimization problems. Whereas the first two applications are characterized by an adaptive setting of the connection strengths and pattern completion the optimization problems are solved with a problem-dependent fixed set of connections and the solutions are given by fixed points of the corresponding dynamical system. The borderline beteen these two application categories is of course vague and inference problems often fall somewhere in between. The rules to be satisfied are implemented by a fixed set of weights like in an optimization problem and the system can be set up to settle with some nodes clamped (pattern completion).

One example of an inference problem is that of scheduling of events in space and time given certain constraints. We have chosen to study this problem is some detail with the neural network approach since some questions with regard to encoding and dynamical properties are particularly transparent in this case. Furthermore this example illustrates a class of industrial applications, which could profit immensely from the parallelism inherent in the neural network formulation.

The scheduling problem has much in common with combinatoric optimization problems like TSP which was mapped onto a neural network in the pioneering work by Tank and Hopfield [1]. In ref. [1] 10- and 30-city problems were studied with good results for the 10-city case. In ref. [2] further studies of the Tank-Hopfield approach are made with respect to refinements and extension to larger problem sizes. The authors of ref. [2] find the results discouraging. The observed problem is related to a redundancy in solution space originating from the neuron multiplexing encoding used in ref. [1]. In ref. [3] this problem was remedied by replacing the neuron multiplexing by multi-state neurons, implying that the simple sigmoidal map, corresponding to the mean field theory equations

$$V_{ia} = \tanh(U_{ia}) \tag{1}$$

of the neuron multiplexing approach, be replaced by a probabilistic map

$$V_{ia} = \frac{e^{U_{ia}}}{\sum_b e^{U_{ib}}} \tag{2}$$

where "i" and "a" denotes city identity and tour number respectively and the local fields U_{ia} are given by

$$U_{ia} = -\frac{1}{T}\frac{\partial E}{\partial V_{ia}} \tag{3}$$

This **multi-state encoding**[1] scheme corresponds to a Potts spin model [4], which was first introduced for solving graph optimization problems in [5].

Also in ref. [3] an automized procedure for locating the position of the critical temperature T_c was developed by making a linear expansion around the trivial fixed point of eq. (2). With this alternative encoding and refinements very good solutions were found in a parameter insensitive

[1]In ref. [3] the somewhat ambiguous term "graded neuron" was used.

way for fairly large problems (200 cities). In ref. [3] the performance of the multi-state encoding algorithm is compared with other more conventional approaches to TSP.

The constraints in TSP are of two kinds. The **hard constraints** are that each city should be visited exactly once and that the tour should be a closed loop. Minimizing the tour length represents a **soft constraint**. In scheduling problems one has a corresponding situation with a set of hard constraints forbidding collisions in space and time and then a set of soft constraints promoting various preferences.

The philosophy of this work is similar to the way we treated TSP [3]. We use the reduced multi-state encoding and estimate T_c in advance. An additional ingredience is that we utilize the power of diagonal **autobias** terms to ensure stable high quality solutions.

This paper is organized as follows: In Sect. 2 we define the problem and map it onto a neural network. Sect. 3 contains a derivation of the phase transition temperature T_c and a discussion of the relevance of the autobias terms for the solution of the MFT equations. Numerical explorations can be found in Sect. 5. In Sect. 6 we close with a brief summary and some concluding remarks.

2 Mapping Scheduling Problems onto a Neural Network

2.1 The Problem

Consider the following scheduling problem: N_p teachers are supposed to have N_q classes in N_x class rooms at N_t time slots. The problem is to find a solution where all N_p teachers give a lecture to each of the N_q classes, using the available space-time slots with no conflicts in space (class rooms) or time. These are the **hard constraints** that have to be satisfied. In addition, one could imagine having a set of **soft constraints** like preferences in time slots, continuity in class-rooms, etc. This is of course just one example of a scheduling problem but we feel it is general and difficult enough to serve as a testbed for the neural network approach.

The basic entities of this problem can be represented by four sets consisting of N_p,N_q,N_x and N_t elements respectively. In what follows, we shall often use the notation $\vec{N} = (N_p, N_q, N_x, N_t)$ for the size parameters of a problem. One could then imagine formulating the problem in terms of neurons connecting two or more elements in different sets. However, there is a more transparent way to desribe the problem that naturally lends itself to the Potts neural encoding of ref. [3]. Consider **events**, defined by teacher-class pairs (p, q), to be mapped onto **space-time slots** (x, t) (see fig. 1). The hard constraints in this picture are as follows:

1. An event (p, q) should occupy precisely one space-time slot (x, t).
2. Different events (p_1, q_1) and (p_2, q_2) should not occupy the same space-time slot (x, t).
3. A teacher p should have at most one class at a time.
4. A class q should have at most one teacher at a time.

A schedule fulfilling all the hard constraints is said to be **legal**.

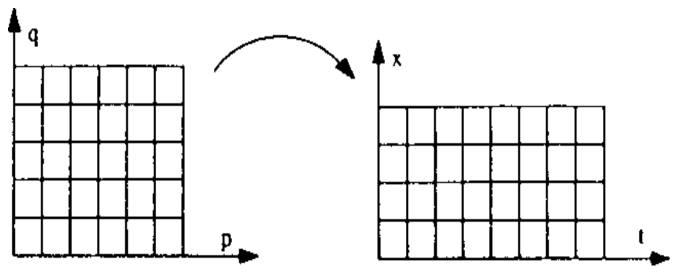

Figure 1: Mapping of events (p,q) onto space-time slots (x,t)

2.2 Neural Network Encoding

The first constraint can be imbedded in a neural network in terms of multi-state neurons $\vec{S}_{pq}$ with components

$$S_{pq;xt} = 0,1 \tag{4}$$

by demanding

$$\sum_{x,t} S_{pq;xt} = 1 \tag{5}$$

for each event (p,q). In other words we have N_pN_q neurons, each of which has N_xN_t possible states . The other three constraints are implemented using energy penalty terms as follows:

$$E_{xt} = \frac{1}{2}\sum_{x,t}\sum_{p_1,q_1}\sum_{p_2,q_2} S_{p_1q_1;xt}S_{p_2q_2;xt} = \frac{1}{2}\sum_{x,t}[\sum_{pq} S_{pq;xt}]^2 \tag{6}$$

$$E_{pt} = \frac{1}{2}\sum_{p,t}\sum_{q_1,x_1}\sum_{q_2,x_2} S_{pq_1;x_1t}S_{pq_2;x_2t} = \frac{1}{2}\sum_{pt}[\sum_{qx} S_{pq;xt}]^2 \tag{7}$$

$$E_{qt} = \frac{1}{2}\sum_{q,t}\sum_{p_1,x_1}\sum_{p_2,x_2} S_{p_1q;x_1t}S_{p_2q;x_2t} = \frac{1}{2}\sum_{qt}[\sum_{px} S_{pq;xt}]^2 \tag{8}$$

From eqs. (4) and (5) it is obvious that any linear combination X of different components of a neuron also must be zero or one, so that

$$X^2 = X \tag{9}$$

For the particular combinations $S_{pq;xt}$, $\sum_x S_{pq;xt}$ and $\sum_t S_{pq;xt}$, this property can be used to add trivial-valued auxiliary terms to the energy:

$$E_{aux} = -\frac{\beta}{2}\sum_{p,q}\sum_{x,t} S^2_{pq;xt} - \frac{\beta_x}{2}\sum_{p,q}\sum_{x,t_1,t_2} S_{pq;xt_1}S_{pq;xt_2} - \frac{\beta_t}{2}\sum_{p,q}\sum_{x_1,x_2,t} S_{pq;x_1t}S_{pq;x_2t} \tag{10}$$

These are the only non-trivial terms of this kind, which also respect the obvious permutation symmetries of the problem. The effect of these extra terms on the energy value is merely that of adding a fixed constant, but they will turn out important for the mean field dynamics.

The problem is now that of finding a configuration that minimizes the total energy,

$$E = E_{xt} + E_{pt} + E_{qt} + E_{aux} + \frac{1}{2}N_pN_q(-3+\beta+\beta_x+\beta_t), \tag{11}$$

where we have added a constant in order for the minimal energy value to be zero.

Our way of inplementing the constraints is by no means unique. One could of course use a different multi-state choice such that there are two sets of $N_p N_q$ neurons with N_x and N_t states respectively. Denoting them $X_{pq;x}$ and and $T_{pq;t}$ the conditions corresponding to eq. (5) read $\sum_x X_{pq;x} = 1$ or $\sum_t T_{pq;t} = 1$. Also different forms of energy-penalty terms are possible. We have explored these other possibilities to some extent and have found that our choice above gives the best performance.

3 Mean Field Dynamics

Now we introduce the continuous **mean field** variables $\vec{V}_{pq}$, corresponding to $\vec{S}_{pq}$. Substituting V for S in the energy expressions, the mean field equations at a finite **temperature** T, as explained in detail in ref. [3], are given in terms of the **local fields** $\vec{U}_{pq}$,

$$U_{pq;xt} = -\frac{1}{T} \cdot \frac{\partial E}{\partial V_{pq;xt}} \tag{12}$$

as

$$V_{pq;xt} = \frac{e^{U_{pq;xt}}}{\sum_{xt} e^{U_{pq;xt}}}. \tag{13}$$

These equations are the analogues of eqs. (2) and (3) for TSP. The natural mean field dynamics for this system consists of iterating eqs. (12-13), and there are two main ways of doing this:

- **Synchronous updating:** A sweep consists of first computing all the local fields, and then updating all the neurons.
- **Serial updating:** Here, for each neuron (p, q), the local field is computed immediately before the corresponding neuron state is updated. Thus, in this case, the sweep order might matter slightly for the outcome of the dynamics. We have normally used ordered sweeps.

For both methods, in addition, the performance depends on the values of the parameters β, β_x, β_t and T. Thus, the next problem is to understand their effect, and to provide them with suitable values.

3.1 Choice of parameters

One can very well run the algorithm at a suitable fixed temperature. Empirically, however, a faster way to obtain good solutions to this kind of problem turns out to be **annealing**, i.e. starting at a high temperature, and successively lowering it in the course of the process. One can get a good understanding of the parameter-dependence by considering the two limits:

- **High T.** At a sufficiently high temperature, the only stable fixpoint will be the totally symmetric state $V^{(0)}_{pq;xt}$, where every neuron component has the constant value

$$V^{(0)}_{pq;xt} = \frac{1}{N_x N_t} \tag{14}$$

The behaviour of the system in this phase is conveniently understood in terms of the linearized dynamics in the neighbourhood of the fixpoint.

- **Low T.** When the temperature is sufficiently low, the system enters another phase: the dynamics effectively becomes discrete, turning into a modified version of **local optimization**.

We shall investigate the two temperature limits in some detail, in particular for the serial updating method, as to the parameter dependence of the performance. First we need a few definitions.

As in ref. [3] we use the **saturation** Σ

$$\Sigma = \frac{1}{N_p N_q} \sum_{pq} \sum_{xt} V_{pq;xt}^2 \tag{15}$$

as a monitor for the freezing process between the high and low temperature limits. In our case it turns out to be convenient also to follow the corresponding sub-saturations Σ_t and Σ_x.

$$\Sigma_t = \frac{1}{N_p N_q} \sum_{pq} \sum_{t} \left(\sum_{x} V_{pq;xt} \right)^2 \tag{16}$$

$$\Sigma_x = \frac{1}{N_p N_q} \sum_{pq} \sum_{x} \left(\sum_{t} V_{pq;xt} \right)^2 \tag{17}$$

The limit $T \rightarrow 0$ is characterized by $\Sigma, \Sigma_t, \Sigma_x \rightarrow 1$.

We also need a an indicator for the rate of change of the configuration: define Δ as the r.m.s. of the neuron component change in a sweep:

$$\Delta = \sqrt{\frac{1}{N_p N_q N_x N_t} \sum_{pqxt} \left(V_{pq;xt}^{(n)} - V_{pq;xt}^{(n-1)} \right)^2} \tag{18}$$

3.2 The High Temperature Limit

Paralleling the analysis of ref. [3], we define T_c as the temperature where the symmetry point turns unstable. Consider the deviation ϵ, defined by

$$V_{pq;xt} = \frac{1}{N_x N_t} + \epsilon_{pq;xt} \tag{19}$$

For small deviations the dynamics, eqs. (12-13), can be linearized and simplifies to

$$\epsilon'_{pq;xt} = \frac{1}{T N_x N_t} \left(\delta_{pq;xt} - \frac{1}{N_x N_t} \sum_{x't'} \delta_{pq;x't'} \right) \tag{20}$$

with

$$\delta_{pq;xt} = \beta \epsilon_{pq;xt} + \beta_x \sum_{t'} \epsilon_{pq;xt'} + \beta_t \sum_{x'} \epsilon_{pq;x't} - \sum_{p'x'} \epsilon_{p'q;x't} - \sum_{q'x'} \epsilon_{pq';x't} - \sum_{p'q'} \epsilon_{p'q';xt} \tag{21}$$

The stability of the symmetric fix-point depends on the size of the dominant eigenvalue of the (sweep) update matrix. If this is smaller than unity in absolute value, the fix-point is stable; otherwise not. The update matrix can be partly diagonalized in terms of a normal-mode expansion in x and t. The temperature, where the transition to instability occurs is thus given by

$$T_c = \max(T_x, T_t, T_{xt}) \tag{22}$$

where T_x, T_t and T_{xt} are temperatures related to modes non-uniform in x, t or both (the completely uniform modes always have eigenvalue zero). Which one is the larger of the three determines which kind of mode turns unstable first. For serial updating, on which we shall concentrate, one obtains

$$\begin{aligned}
T_x &= \frac{1}{N_x N_t}\left(\frac{1}{2} + \left|\beta - \frac{1}{2} + N_t\beta_x\right|\right) & (23)\\
T_t &= \frac{1}{N_x N_t}\left(\frac{1}{2} + N_x + \left|\beta - \frac{1}{2} + N_x\left(\beta_t - 1\right)\right|\right) & (24)\\
T_{xt} &= \frac{1}{N_x N_t}\left(\frac{1}{2} + \left|\beta - \frac{1}{2}\right|\right) & (25)
\end{aligned}$$

For completeness, we also give the corresponding result for synchronous updating:

$$\begin{aligned}
T_x^{(syn)} &= \frac{1}{N_x N_t}\left(\frac{N_p N_q}{2} + \left|\beta + N_t\beta_x - \frac{N_p N_q}{2}\right|\right) & (26)\\
T_t^{(syn)} &= \frac{1}{N_x N_t}\left(\frac{N_x N_p + N_x N_q + N_p N_q}{2} + \left|\beta + N_x\beta_t - \frac{(N_x N_p + N_x N_q + N_p N_q)}{2}\right|\right) & (27)\\
T_{xt}^{(syn)} &= \frac{1}{N_x N_t}\left(\frac{N_p N_q}{2} + \left|\beta - \frac{N_p N_q}{2}\right|\right) & (28)
\end{aligned}$$

It should be stressed, that the above expressions for T_c are *exact*, in contrast to the case of TSP, where only estimates based on averages and variances of the city-distance matrix could be made [3].

We have thus established the parameter dependence of both the critical temperature and the relevant mode, for the transition from the symmetric phase. It turns out that, for modest parameter-values, the pure t-mode will be the first to turn unstable. This means, loosely speaking, that the events (p, q) will first decide on which time-slot t to use, and only after that, they will start worrying about the class-rooms x. This situation can of course be changed by modifying the parameters; we will return to this point below.

3.3 The Low Temperature Limit

When the annealing proceeds to low temperatures, the updating equation (13) effectively reduces to

$$V_{pq;xt} = \delta_{x,x_{pq}}\delta_{t,t_{pq}} \tag{29}$$

with x_{pq} and t_{pq} the values of x and t that maximize $U_{pq;xt}$. We note that, as far as the neuron (p, q) is concerned, keeping the other neurons fixed, the corresponding local field $\vec{U}_{pq}$ can be written as

$$U_{pq;xt} = -1/T\left(E_{xt}^{(p,q)} - (\beta - 1)\delta_{x,x_0}\delta_{t,t_0} - \beta_x\delta_{x,x_0} - (\beta_t - 2)\delta_{t,t_0} + const.\right) \tag{30}$$

where $E_{xt}^{(p,q)}$ is the energy value if the state (x,t) is chosen, (x_0,t_0) is the present state of the neuron, and the constant compensates for the part of the energy that depends on th other neurons only. The three δ-terms come from the diagonal (in (p,q)) part of the quadratic energy expression (11), i.e. the terms that couple the neuron to itself.

For the particular choice of parameters

$$\beta = 1 \tag{31}$$
$$\beta_x = 0 \tag{32}$$
$$\beta_t = 2 \tag{33}$$

the energy expression contains no terms diagonal in (p,q)[2]. We conclude from eq. (30), that in this case maximizing $U_{pq;xt}$ is equivalent to minimizing the energy. In effect, eq. (29) will correspond to **local optimization** of the energy. This is precisely what we expect in the low-temperature limit.

For all other parameter-values, we instead obtain what could be called **autobiased local optimization**. What is minimized is an effective energy,

$$E_{eff} = E - \gamma \tag{34}$$

where γ depends only on the diagonal part of the energy expression. It corresponds to a bias with respect to the present neuron state, and for a given neuron (p,q) it is given by

$$\gamma = (\beta - 1)\sum_{xt} V_{pq;xt}V_{pq;xt}^{(0)} + \beta_x \sum_{xtt'} V_{pq;xt}V_{pq;xt'}^{(0)} + (\beta_t - 2)\sum_{xx't} V_{pq;xt}V_{pq;x't}^{(0)} \tag{35}$$

with $\vec{V}^{(0)}$ denoting the present state of the neuron. At low temperature, γ can be expressed as

- $\gamma = \gamma_0 \equiv (\beta - 1) + \beta_x + (\beta_t - 2)$, for staying in the present state.
- $\gamma = \gamma_x \equiv \beta_x$, for changing t only.
- $\gamma = \gamma_t \equiv \beta_t - 2$, for changing x only.
- $\gamma = 0$, otherwise.

These **autobias** terms in the energy correspond to the coupling of neurons to themselves (see fig. 2), such that the continuity in (computer) time of the neurons is either rewarded or penalized, depending on the values of the connection strengths[3]. One effect of the autobias terms is that they affect the possibility of escaping from local minima. Since the energy at zero temperature is quantized in units of unity, a choice of parameters such that the difference between any pair of γ's above is less than unity (we will refer to this case as **low autobias**), will lead to a low temperature limiting behaviour similar to the case without autobias, eqs. (31-33). The behaviour at a non-zero temperature will be different, however. Within the low-autobias region, one can choose the parameters such, that unwanted local minima become flattened out, while at the same time keeping the difference in critical temperatures between the different modes reasonably low.

[2]It is thus multi-linear in the neurons, which is precisely the case where our mean field equations (12-13) coincide with the traditional mean field notion [6].

[3]Autobias is not to be confused with what is normally called bias, and which corresponds to a connection to a permanently fixed, external neuron.

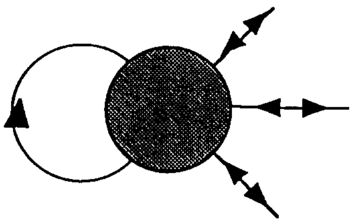

Figure 2: Schematic view of a neuron with a self-coupling corresponding to a diagonal term in the energy.

Empirically, we find the parameter choice corresponding to

$$\gamma_0 = \gamma_x = \gamma_t = -O(0.5 - 1) \tag{36}$$

to be close to optimal for "dense" problems, i.e. when $N_p N_q = N_x N_t$. This choice corresponds to a preference for changing both x and t. For more sparse problems, which are easier due to the large phase space, the choice

$$\gamma_0 = \gamma_x = 0, \gamma_t = -O(0.5 - 1) \tag{37}$$

turns out to be more appropriate.

4 Numerical Studies

Our algorithm is implemented in the following "black box" manner in the case of serial updating:

- Choose $\gamma_0 = \gamma_x = \gamma_t = -0.8$ for a dense problem and $\gamma_0 = \gamma_x = 0, \gamma_t = -0.8$ for a sparse problem (cf. eqs. (36,37)).
- Determine T_c according to eqs. (22, 23, 24, 25).
- Initialize with $V_{ia} = 1/K + 0.001 \times rand[-1,1]$.
- Anneal according to $T_n = 0.95 \times T_{n-1}$ with one sweep per temperature unless $\Delta\Sigma/\Sigma > 0.05$, in which case $T_n = T_{n-1}$.
- Stop iterating when $\Sigma \geq 0.999$, or $\Sigma > 0.8$ and $\Delta < 0.01$.
- If the final solution is non-legal, in the sense that one of the conditions 2-4 in Sec. 2.1 is not fulfilled, redo the steps above with a different random seed.

This procedure differs slightly from what was used in ref. [3] for the graph partition and travelling salesman problems. The scheduling problem requires a slightly more delicate annealing. The schedule above feels the onset of the phase transition region, since $\Delta\Sigma/\Sigma$ is a rough measure of the entropy change. Also we do not apply any greedy heuristic in the case of non-legal final solutions since in the scheduling case no simple and fast such procedure exists. Rather we reheat the system and let it cool down again until a legal solution has been obtained.

We have investigated the performance of this algorithm for a variety of problem sizes and for different levels of difficulty. With a difficult problem we mean that the number of available space-time slots is small compared to the number of events. There are three obvious measures of the competition for space-time slots defined as follows:

$$\eta_{pq} = \frac{N_p N_q}{N_x N_t} \tag{38}$$

$$\eta_p = \frac{N_p}{N_t} \tag{39}$$

$$\eta_q = \frac{N_q}{N_t} \tag{40}$$

In other words the difficulty increases with η_{pq}, η_p and η_q. In what follows we denote problems in the limiting case $\eta_{pq}, \eta_p, \eta_q \to 1$ **dense problems**. Correspondingly $\eta_{pq}, \eta_p, \eta_q < 1$ corresponds to **sparse problems**.

4.1 Hard Constraints

Let us now consider problems with hard constraints (1-4) only, considering the dense and sparse cases separately.

Dense Problems

We have applied our algorithm to dense problems ($\eta_p = \eta_q = \eta_{pq} = 1$) with sizes $(N_p, N_q) = (5,5)$,...,$(12,12)$. In fig. 3a we show the average number of trials needed to obtain a legal solution for an experiment. The corresponding average convergence times are shown in fig. 3b. With convergence time we mean the total number of sweeps needed to obtain a legal solution no matter how many trials it takes. The error bars in fig. 3, as well as in forthcoming figures of the same type were obtained by dividing the standard deviation by the square-root of the number of experiments.

It is clear from fig. 3 that the algorithm consistently finds a legal solution more or less at the first attempt independently of problem size. No size dependence of the convergence time is observable (note that a 12×12 scheduling problem corresponds to a 144-city TSP problem in terms of the number of degrees of freedom).

It is illuminating to study how the subenergies (E_{pt},E_{qt},E_{xt}), the saturations (Σ_x,Σ_t,Σ_{xt}) and the temperature T vary with the number of iterations N_{sw}. This is shown in fig. 4 for an $\vec{N} = (7,7,7,7)$ run, in which the symmetrical high temperature phase clearly breaks up in two steps; first in t and later in x.

Sparse Problems

Next we release the difficulty by considering the case $\eta_p = \eta_q = \eta_{pq} \approx 0.8$, again with $(N_p, N_q) = (5,5), ..., (12,12)$. In this case there are more available space-time slots than there are events. Hence one would not expect that the MFT algorithm will always converge to a fixed point representing a single particular choice. Instead it might converge into a linear combination of different legal solutions. Indeed, this turns out to be the case very often. For such mixed states, the saturation

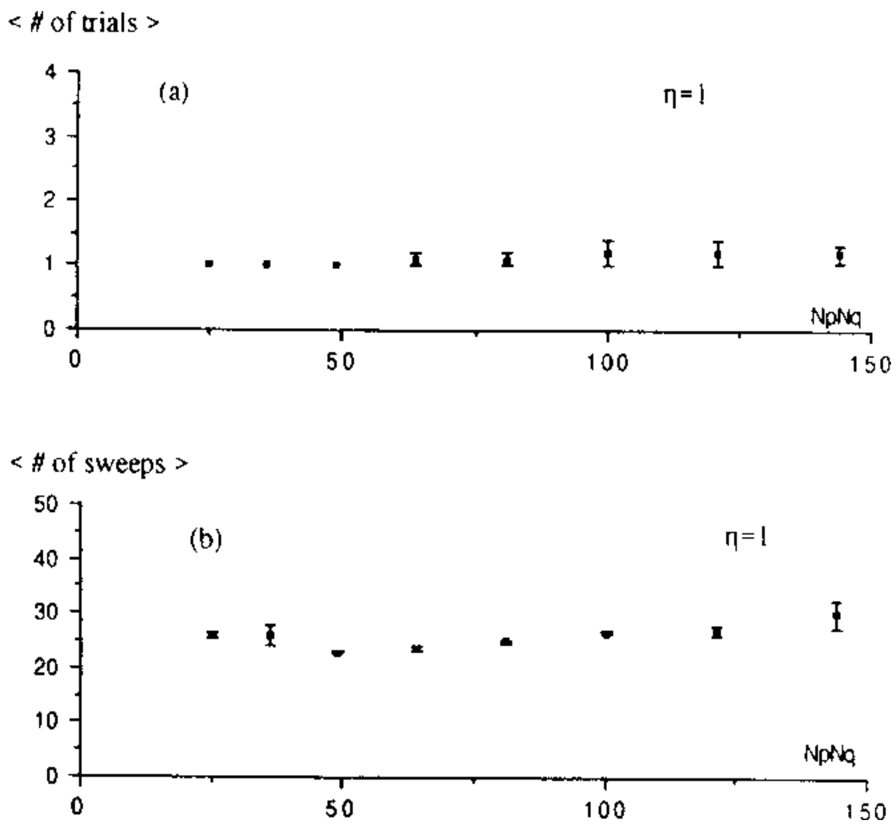

Figure 3: (**a**) Average number of trials needed to obtain a legal solution as a function of number of events $(N_p N_q)$ for $\eta = 1$. (**b**) The corresponding average convergence time. All data points are based on 10 experiments.

Σ never turns to one. Accordingly, for sparse problems we have modified the convergence criteria from $\Sigma \geq .999$ to $\Sigma > .8$ and $\Delta < 0.01$. In fig. 5 we show performance and convergence times for $\eta_{pq} \approx 0.8$ problems. Again we conclude that the performance of the algorithm is excellent with almost no deterioration as the size increases. The convergence times are slightly higher than in the dense case. This is not surprising, considering the increased degeneracy of solution-space in the sparse case.

In table 1 we show a mixed state solution for an $\vec{N} = (5, 5, 5, 7)$ problem, where an appealing feature is obvious: Part of the decision is left open to the user in such a way that, whatever choice is made, the resulting solution is legal.

4.2 Hard and Soft Constraints

Next we want to add additional constraints to the the energy functions, which are "soft" in their nature, in the sense that the system should try to accomodate to them as much as possible but violations do not lead to non-legal configurations. Adding soft constraints lifts some of the degeneracy

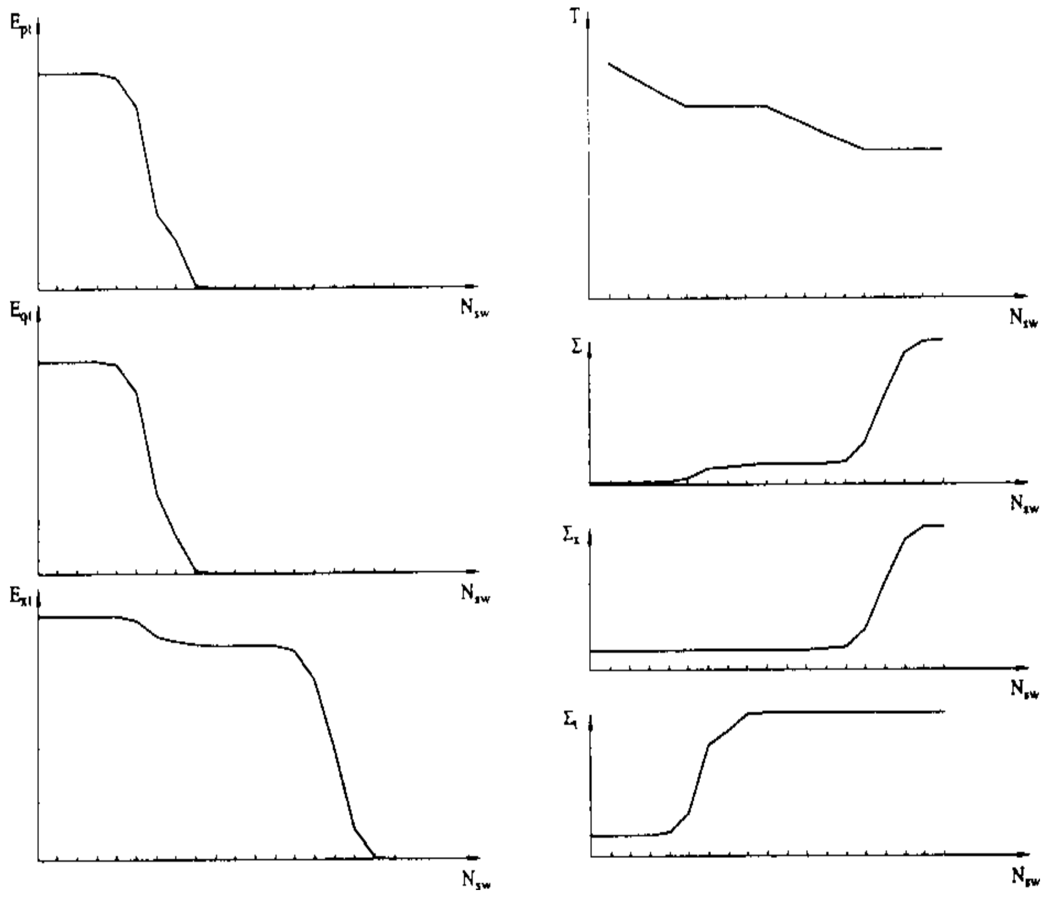

Figure 4: E_{pt}, E_{qt}, E_{xt}, T, Σ, Σ_x and Σ_t vs. number of iterations N_{sw} for an $\vec{N} = (7,7,7,7)$ problem. (The energies shown are defined without diagonal terms, which turns out to be more natural.)

implicit in the pure hard constraints problem and would be expected to make the algorithm having easier to decide between otherwise equally good solutions. One also expects that dense problems would be more sensitive to such a symmetry breaking than sparse problems where the existence of empty space-time slots increases degeneracy.

An example of a soft constraint that is easy to implement, yet not trivial, is to require that a class should remain in the same room from one time-slot to the next (defining as a **bond**). This constraint might be encoded by including an additional energy term:

$$E_\alpha = -\frac{\alpha}{2} \sum_{p' \neq p} \sum_q \sum_{xt} S_{pq;xt} \left(S_{p'q;x(t+1)} + S_{p'q;x(t-1)} \right) \tag{41}$$

where $t \pm 1$ is defined modulo N_t. The strength of this term is controlled by the **adhesion** α, which by definition is never allowed to be large enough to dominate the hard constraints and thus should

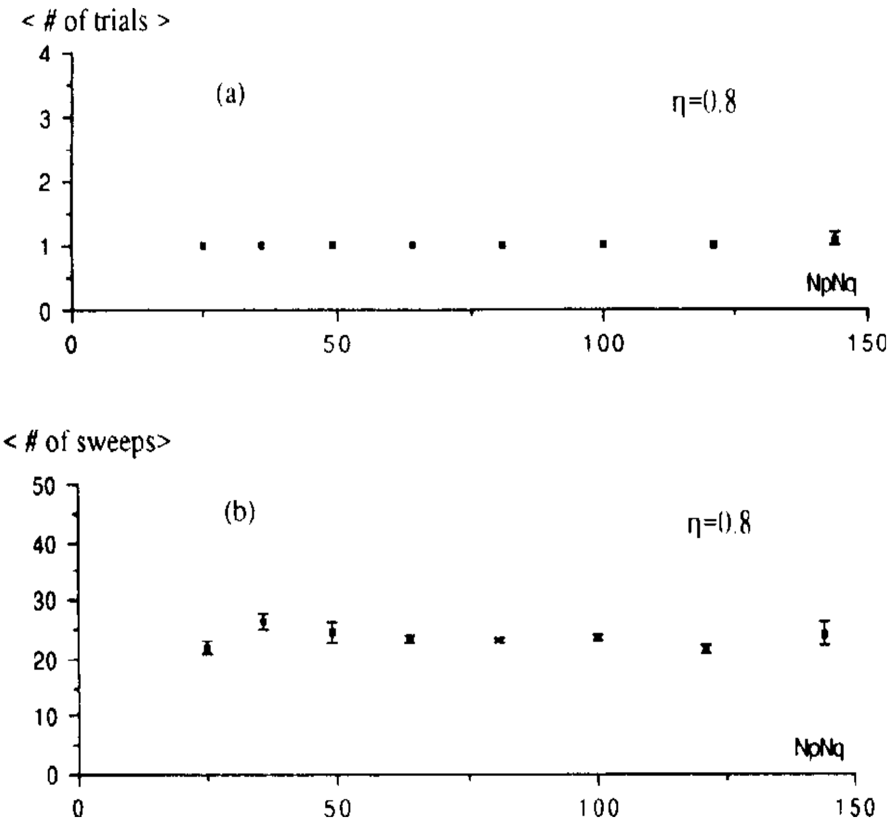

Figure 5: **(a)** Average number of trials needed to obtain a legal solution as a function of number of events (N_pN_q) for $\eta \approx 0.8$. **(b)** The corresponding average convergence time. All data points are based on 10 experiments.

be $< O(1)$. Defining the **bonding** as

$$B = \frac{\#\ of\ bonds}{maximal\ \#\ of\ bonds} \tag{42}$$

we investigate the relation between the adhesion and the resulting bonding [4]. In fig. 6, B is shown as a function of α for some dense testbeds. It is clear that even moderate values of α (0.2) give satisfactory bonding. Note the possibility to have negative adhesion, forcing the bonding towards zero. The average number of trials needed to obtain a legal solution increased slightly with increasing problem size. By tuning the parameter values, this increase in convergence time can be eliminated. We have, however, chosen to display all the results with a fixed choice of γ, γ_x and γ_t in order to demonstrate the parameter insensitivity.

The corresponding behaviour for sparse problems is shown in fig. 7. As can be seen the results are very similar to the dense case with a slightly smoother α-dependence. The most interesting thing is that the number of sweeps needed to reach a good solution decreases with increasing adhesion in

[4]Bonds are counted modulo t, but in a dense problem we do not count the bond that closes a cyclic sequence in t. The maximum number of bonds is thus always $N_q(N_p - 1)$.

	x				
	1	2	3	4	5
t 1	2/4	3/1	5/3	1/5	4/2
2	(2/2)	1/1	5/4	(4/5)	(3/3)
3	-	-	-	-	-
4	1/2	(5/1)	3/4	(4/5)	2/3
5	(1/4)	4/1	2/5	5/2	(3/3)
6	(5/1)	4/3	(1/4)	3/5	(2/2)
7	4/4	1/3	5/5	2/1	3/2

Table 1: Example of a resulting mixed-state solution for an $\vec{N} = (5, 5, 5, 7)$ problem. The entries represent p/q-combinations. Entries within parentheses appear twice, and represent split decisions.

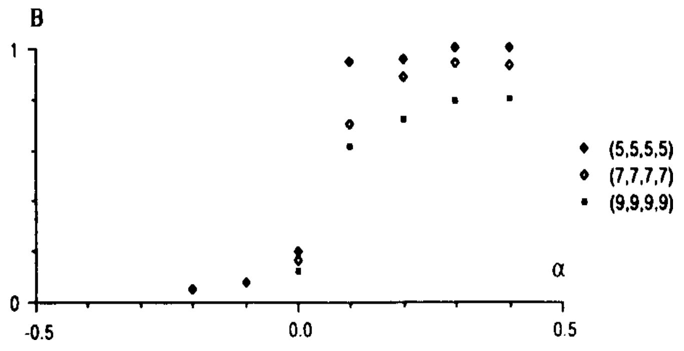

Figure 6: Bonding as a function of α for $\vec{N}$=$(5, 5, 5, 5)$, $(7, 7, 7, 7)$ and $(9, 9, 9, 9)$ respectively.

accordance with the expected decreasing degeneracy. Imposing soft constraints will in this respect improve the performance of the algorithm.

In figs. 8 and 9 the results corresponding to figs. 3 and 5, but including the adhesion constraint with $\alpha = 0.2$, are shown.

5 Summary and Outlook

We have demonstrated that the neural network paradigm is an efficient, reliable and potentially concurrent approach for finding good solutions to scheduling problems. It would be interesting to make performance comparisons with other "non-neural" approaches (if available) like the Kernighan-Lin algorithm [8], which has a state-of-the-art reputation for the TSP [9]. Our numerical investigations were limited to "teachers and classes" problems with maximal size $\vec{N} = (12, 12, 12, 15)$. Almost no signs of deteriorating solution quality with size were noticed, so we feel confident that the observed quality of the solutions will survive up to realistic problem sizes (50×50). (We intend to investigate larger problems in connection with more intense studies of soft constraints [7].) Most efforts in the numerical studies were spent on problems with entirely hard constraints (see Sec. 2.1), but the inclusion of soft constraints like having classes in a row in the same classroom did not significantly lower the solution quality. The key ingredients that were important for the success are the following:

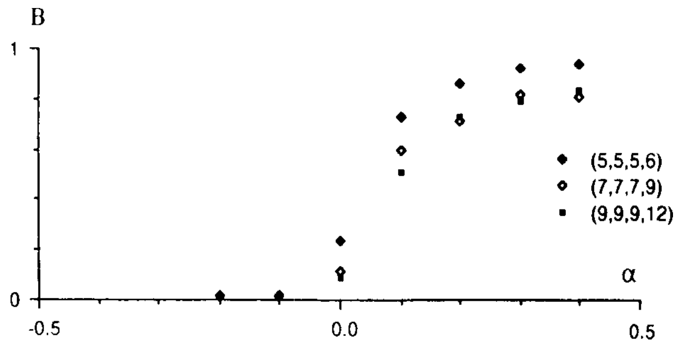

Figure 7: Bonding as a function of α for $\vec{N}$=(5, 5, 5, 6), (7, 7, 7, 9) and (9, 9, 9, 12) respectively.

- **Multi-state neuron encoding**. We used the reduced encoding of eq. (5). As in the case of the graph partition and travelling salesman problems [3] this reduction of solution space removes destructive redundancy when the MFT equations settle.

- **Advance estimate of T_c**. In order to have a "black box" algorithm one needs to estimate the temperature where the crucial phase transition takes place. This is done by a linear expansion around the high temperature fixed point as in ref. [3].

- **Auxiliary energy terms**. Of particular importance for this kind of problem is the employment of auxiliary terms in the energy function, which are redundant with respect to the encoding but play a very important dynamical role for avoiding getting stuck in local minima and for stabilizing good solutions. Their presence were found to be important also for TSP [3]. Here they are even more crucial since in the "teachers and classes" problem one has an asymmetry in x and t giving rise to two different phase transitions. With a suitable choice of auxiliary energy terms the distance between the phase-transitions can be decreased, which is important in order to speed up the settling.

We end this section with some comments and a brief mentioning of additional virtues and possible extensions of our approach, some of which are presently being investigated [7].

- **Clamping**. The hard constraints in our testbeds have been limited to the four "legal" requirements listed in Sect. 2.1. A typical application will also contain additional hard constraints in terms of some precise requirements like "teacher p' has to lecture class q' in room x' at time t'". This is of course easily implemented in our algorithm by clamping $S_{p'q';xt}$ to the value $\delta_{x,x'}\delta_{t,t'}$ throughout the MFT annealing. We have performed experiments of this kind with good results, which is not surprising since clamping breaks symmetries and hence facilitates the process of finding good solutions rapidly.

- **Pattern Completion**. With several events clamped the settling process represents an example of pattern completion. It has the flavour of both a strict combinatorial optimization problem like TSP with dynamics set by the weights and a feature recognition or content-addressable memory application where the outcome is also influenced by the initially clamped state.

- **Fuzzy logic**. The pseudo-stochastic or MFT technique for finding solutions to the scheduling problem brings an additional advantage to the scene. The neuronic MFT variables being

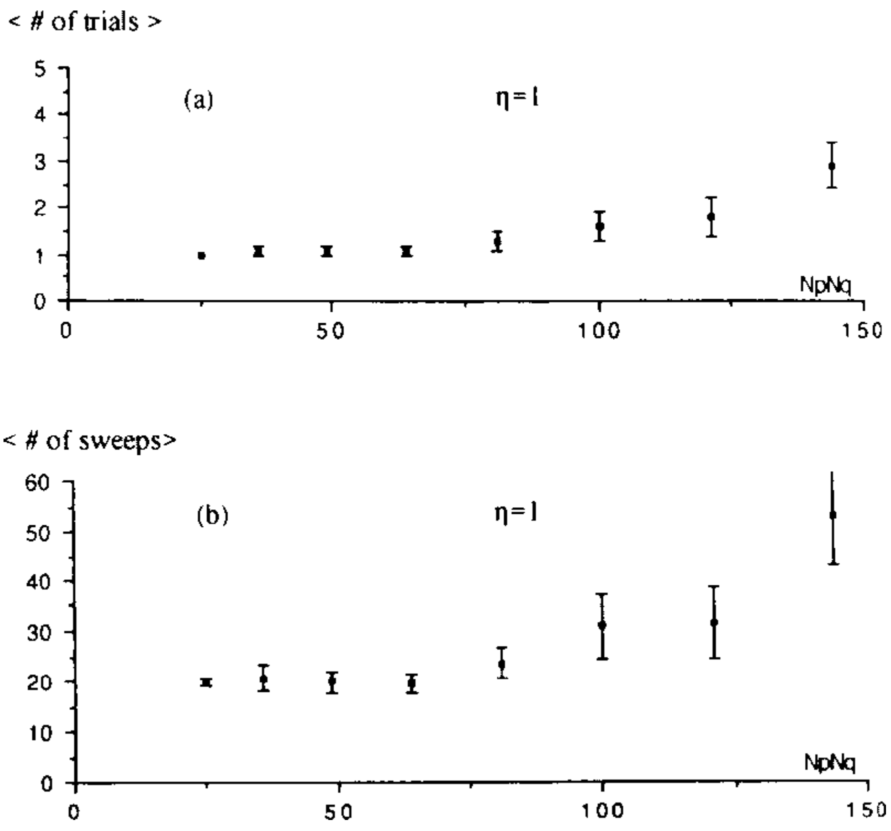

Figure 8: **(a)** Average number of trials needed to obtain a legal solution as a function of number of events $(N_p N_q)$ for $\eta = 1$ with adhesion $\alpha = 0.2$. **(b)** The corresponding average convergence time. All data points are based on 10 experiments.

probabilities makes it possible to specify initial conditions (clamping) in cases where some uncertainty exists.

- **Revision capability**. Many real-world scheduling applications demand the possibility of performing revisions after parts the solution has been implemented. Such an interactive constraint satisfaction process is natural within the neural network framework. As time goes by one clamps those values of $S_{pq;xt}$ corresponding to events that have taken place and redo the MFT settling with the revised constraints.

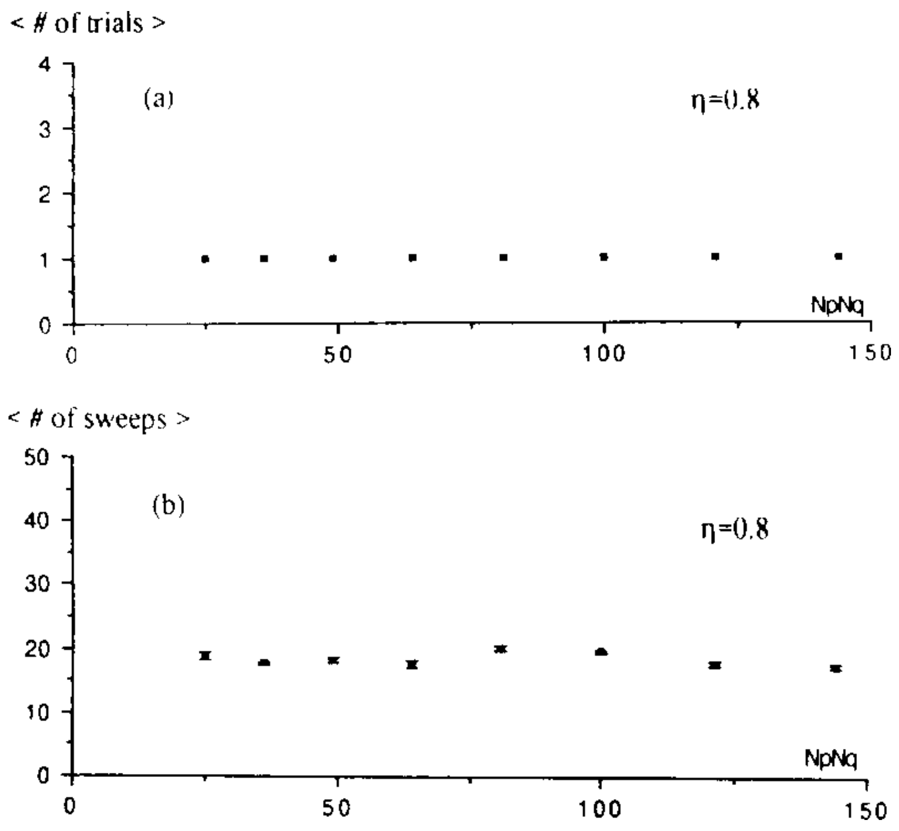

Figure 9: (**a**) Average number of trials needed to obtain a legal solution as a function of number of events (N_pN_q) for $\eta \approx 0.8$ with adhesion $\alpha = 0.2$. (**b**) The corresponding average convergence time. All data points are based on 10 experiments.

References

[1] J.J. Hopfield and D.W. Tank, "Neural Computation of Decisions in Optimization Problems", *Biological Cybernetics* **52**, 141 (1985).

[2] G.V. Wilson and G.S. Pawley, "On the Stability of the Travelling Salesman Problem Algorithm of Hopfield and Tank", *Biological Cybernetics* **58** 63 (1988).

[3] C. Peterson and B. Söderberg, "A New Method for Mapping Optimization Problems onto Neural Networks", *International Journal of Neural Systems* **1**, 3 (1989).

[4] F.Y. Wu, "The Potts Model", *Review of Modern Physics* **54**, 235 (1983).

[5] I. Kanter and H. Sompolinsky, "Graph Optimization Problems and the Potts Glass", *Journal of Physics* **A 20**, L673 (1987).

[6] H. Flyvbjerg, P. Mansfield and B. Söderberg, "High Precision Mean-Field Results for Lattice Gauge Theories", *Nuclear Physics* **B240** [**FS12**], 171 (1984).

[7] L. Gislén, C. Peterson and B. Söderberg, in preparation.

[8] S. Lin and B.W. Kernighan, "An Efficient Heuristic Algorithm for Solving the Travelling Salesman Problem", *Operations Research* **21**, 498 (1973).

[9] D. Johnson, "More Approaches to the Travelling Salesman Guide", *Nature* **330**, 525 (1087). This reference briefly compares different approaches to the TSP. (It is, however based on somewhat old material as far as neural networks and genetic algorithms go.)